ARCHITECTURAL DESIGNS OF ECHO STATE NETWORK

by

ALI ABDALLAH ALI AL RODAN

A thesis submitted to
The University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Computer Science
College of Engineering and Physical Sciences
The University of Birmingham
May 2012

Abstract

Reservoir computing (RC) refers to a new class of state-space models with a fixed state transition structure (the *"reservoir"*) and an adaptable readout from the state space. The reservoir is supposed to be sufficiently complex so as to capture a large number of features of the input stream that can be exploited by the reservoir-to-output readout mapping. The field of RC has been growing rapidly with many successful applications. However, RC has been criticised for not being principled enough. Reservoir construction is largely driven by a series of randomised model building stages, with both researchers and practitioners having to rely on a series of trials and errors. Echo State Networks (ESNs), Liquid State Machines (LSMs) and the back-propagation decorrelation neural network (BPDC) are examples of popular RC methods. In this thesis we concentrate on Echo State Networks, one of the simplest, yet effective forms of reservoir computing.

Echo State Network (ESN) is a recurrent neural network with a non-trainable sparse recurrent part (reservoir) and an adaptable (usually linear) readout from the reservoir. Typically, the reservoir connection weights, as well as the input weights are randomly generated. ESN has been successfully applied in time-series prediction tasks, speech recognition, noise modelling, dynamic pattern classification, reinforcement learning, and in language modelling, and according to the authors, they performed exceptionally well.

In this thesis, we propose simplified topologies of the original ESN architecture and we experimentally show that a *Simple Cycle Reservoir* (SCR) achieved comparable performances to 'standard' ESN on a variety of data sets of different origin and memory

structure, hence, most tasks modelled by ESNs can be handled with very simple model structures. We also proved that the memory capacity of linear SCR can be made arbitrarily close to the proven optimal value (for any recurrent neural network of the ESN form).

Furthermore, we propose to extend the simple cycle reservoir (SCR) with a regular structure of shortcuts (Jumps) - *Cycle Reservoir with Jumps* (CRJ). In the spirit of SCR we keep the reservoir construction simple and deterministic. We show that such a simple architecture can significantly outperform both the SCR and standard randomised ESN. Prompted by these results, we investigate some well known reservoir characterisations, such as eigenvalue distribution of the reservoir matrix, pseudo-Lyapunov exponent of the input-driven reservoir dynamics, or memory capacity and their relation to the ESN performance.

Moreover, we also design and utilise an ensemble of ESNs with diverse reservoirs whose collective readout is obtained through Negative Correlation Learning (NCL) of ensemble of Multi-Layer Perceptrons (MLP), where each individual MPL realises the readout from a single ESN. Experimental results on three data sets confirm that, compared with both single ESN and flat ensembles of ESNs, NCL based ESN ensembles achieve better generalisation performance.

In the final part of the thesis, we investigate the relation between two quantitative measures suggested in the literature to characterise short term memory in input driven dynamical systems, namely the short term memory capacity spectrum and the Fisher memory curve.

Acknowledgements

I would like to thank my supervisor Dr Peter Tino, for introducing me to this area of research "Reservoir Computing", and for valuable advice and support during my PhD study.

I would also like to thank my thesis group members, Dr Richard Dearden and Dr Iain Styles for sharing their knowledge in the area of Machine Learning and Neural Networks.

Special acknowledgement is given to the examiners of the thesis, Prof. Colin Fyfe and Dr. John Bullinaria, for agreeing to be the examiners of my PhD Viva.

I also thank all the staff in the Department of Computer Science at The University of Birmingham. A great deal of thanks goes to my colleagues: Zaid Al-Zobaidi, Mohammed Wasouf, Saeed Alghamdi, Adnan Alrashid, and Hasan Qunoo. They made the daily grind of being a research student so much fun.

I also wish to express my gratitude to all of the people mentioned and not mentioned above for reading through numerous drafts of this thesis. They will be missed and I wish them all a very happy and successful life and career.

Finaly and most important, I am deeply grateful to my parents, my sister, and my brothers Ahmad, Omar, and Mohammed - for sharing, motivating and inspiring me in good and bad moments.

Contents

List of Figures

List of Tables

List of Abbreviations

RNN ..... Recurrent Neural Network
RC ..... Reservoir Computing
ESN ..... Echo State Network
ESP ..... Echo State Property
LSM ..... Liquid State Machines
BPDC ..... BackPropagation DeCorrelation neural network
FPM ..... Fractal Prediction Machines
SCR ..... Simple Cycle Reservoir
DLR ..... Delay Line Reservoir
DLRB ..... Delay Line Reservoir with feedback connections
CRJ ..... Cycle Reservoir with Jumps
MC ..... Memory Capacity
STM ..... Short-Term Memory
LSTM ..... Long Short-Term Memory
FMC ..... Fisher memory curve
BPTT ..... Backpropogation through time
RTRL ..... Real-Time Recurrent Learning
KF ..... Kalman Filter
EKF ..... Extended Kalman filter
NCL ..... Negative Correlation Learning
FFNN ..... Feedforward Neural Network
MLP ..... Multi-Layer Perceptrons
MSE ..... Mean Square Error
NMSE ..... Normalised Mean Square Error
WER ..... Word Error Rate
i.i.d. ..... Independent and Identically Distributed
SVD ..... Singular value Decomposition
RLS ..... Recursive Least Squares

Cov ..... Covariance
Var ..... Variance
NARMA ..... Non-linear Auto-Regressive Moving Average
WTA ..... Winner-Take-All
IPIX ..... Ice MultiParameter Imaging X-band
tanh ..... Tangent hyperbolic
Log ..... Logistic map

List of Notation

t	current time	1
N	dimension of the state space model (reservoir size)	1
$\mathcal{E}$	cost function	2
W	reservoir weight matrix	3
α	reservoir matrix scaling parameter	4
K and L	input and output units	10
$s(t)$	input data at time t	10
$x(t)$	activation of the reservoir units at time t	10
$y(t)$	output data at time t	10
V and U	input and output weight matrices	10
f	activation function	10
g	nonlinear output function	11
t_{start} and t_{end}	start and end time for an epoch	12
$E(t)$	error of the current output at time t	12
$\hat{y}(t)$	current output at time t	12
f'	derivative of the activation function	13
λ	learning parameter	15
$z(t+1)$	an optional uniform i.i.d. noise at time $t+1$	17
$\lvert\lambda_{max}\rvert$	the spectral radius	18
γ	forgetting parameter for online training using RLS	20
$F_i(t)$	output for network i at time t	21
$F(t)$	ensemble outpu at time t	21
p_i	negative correlation enforcing penalty term for network i	22
L_{trn}, L_{val} and L_{tst}	length of the training, validation and test sequences	29
ℓ	jump size	60
$x^i(t)$	reservoir state for ESN as an input for the MLP readout	90
$J(k)$	fisher memory curve for a delay k	97

Chapter 1

Introduction

A large number of models designed for time series processing, forecasting or modelling follow a *state-space formulation*. At each time step t, all *'relevant'* information in the driving stream processed by the model up to time t is represented in the form of a *state* (at time t). The model output depends on the past values of the driving series and is implemented as a function of the state - the so-called *read-out* function. The state space can take many different forms, e.g. a finite set, a countably infinite set, an interval etc. A crucial aspect of state-space model formulations is an imposition that the state at time $t+1$ can be determined in a recursive manner from the state at time t and the current element in the driving series (*state transition* function). Depending on the application domain, numerous variations on the state space structure, as well as the state-transition/readout function formulations have been proposed.

One direction of research into a data-driven state space model construction imposes a state space structure (e.g. an N-dimensional interval) and a semi-parametric formulation of both the state-transition and readout functions. The parameter fitting is then driven by a cost functional $\mathcal{E}$ measuring the appropriateness of alternative parameter settings for the given task. Recurrent neural networks (RNNs) are examples of this type of approach (Atiya and Parlos, 2000). If $\mathcal{E}$ is differentiable, one can employ the gradient of $\mathcal{E}$ in

the parameter fitting process. However, there is a well known problem associated with parameter fitting in the state-transition function (Bengio et al., 1994): briefly, in order to 'latch' an important piece of past information for the future use, the state-transition dynamics should have an attractive set. In the neighbourhood of such a set the derivatives vanish and hence cannot be propagated through time in order to reliably bifurcate into a useful latching set.

A class of approaches referred to as *reservoir computing (RC)* try to avoid this problem by fixing the state-transition function - only the readout is fitted to the data (Lukosevicius and Jaeger, 2009; Schrauwen et al., 2007b). The state space with the associated state transition structure is called the *reservoir*. The reservoir is supposed to be sufficiently complex so as to capture a large number of features of the input stream that can potentially be exploited by the readout. Echo State Networks (ESNs) (Jaeger, 2001), Liquid State Machines (LSMs) (Maass et al., 2002) and the back-propagation decorrelation neural network (BPDC) (Steil, 2004) are examples of popular RC models.
These models differ in how the fixed reservoir is constructed and what form the readout takes. For example, *echo state networks* (ESN) (Jaeger, 2001) typically have a linear readout and a reservoir formed by a fixed recurrent neural network type dynamics. *Liquid state machines* (LSM) (Maass et al., 2002) also mostly have a linear readout (some cases have Multilayer Feedforward Neural Network (FFNN) readout of spiking or sigmoid neurons) and the reservoirs are driven by the dynamics of a set of coupled spiking integrate-and-fire neuron models. Back-propagation decorrelation neural network (BPDC) (Steil, 2004) is an online RNN learning algorithm uses the idea of Atiya and Parlos efficient version of gradient descent RNN learning algorithm (Atiya and Parlos, 2000) by adapting only the output weights, the input and hidden (reservoir) weights are remain constant. *Fractal prediction machines* (FPM) (Tino and Dorffner, 2001) have been suggested for processing symbolic sequences. Their reservoir dynamics is driven by fixed affine state transitions over an N-dimensional interval. The readout is constructed as a collection of multinomial distributions over next symbols. Many other forms of reservoirs can be found in the liter-

ature (e.g. (Jones et al., 2007; Deng and Zhang, 2007; Dockendorf et al., 2009; Bush and Anderson, 2005; Ishii et al., 2004; Schmidhuber et al., 2007; Ajdari Rad et al., 2008)). However, exactly what aspects of reservoirs are responsible for their often reported superior modelling capabilities (Jaeger, 2001, 2002a,b; Jaeger and Hass, 2004; Mass et al., 2004; Tong et al., 2007) is still unclear. In this thesis we concentrate on Echo State Networks, one of the simplest, yet effective forms of reservoir computing.

Roughly speaking, Echo State Network (ESN) (Jaeger, 2001, 2002a,b; Jaeger and Hass, 2004) is a recurrent neural network with a non-trainable sparse recurrent part (reservoir) and a simple linear readout. Connection weights in the ESN reservoir, as well as the input weights are randomly generated from a uniform distribution.

1.1 Motivation

Echo State Network (ESN) has been successfully applied in time-series prediction tasks (Jaeger and Hass, 2004), speech recognition (Skowronski and Harris, 2006), noise modelling (Jaeger and Hass, 2004), dynamic pattern classification (Jaeger, 2002b), reinforcement learning (Bush and Anderson, 2005), and in language modelling (Tong et al., 2007).

A variety of extensions/modifications of the classical ESN can be found in the literature, e.g. intrinsic plasticity (Schrauwen et al., 2008b; Steil, 2007), refined training algorithms (Jaeger and Hass, 2004), training with Neuroscale (Wang and Fyfe, 2011), leaky-integrator reservoir units (Jaeger et al., 2007a), support vector machine (Schmidhuber et al., 2007), setting the reservoir weights using Self-Organizing Maps (SOM) and Scale-Invariant Maps (SIM) (Basterrech et al., 2011), filter neurons with delay&sum readout (Holzmann and Hauser, 2009), pruning connections within the reservoir (Dutoit et al., 2009) etc. There have also been attempts to impose specialised interconnection topologies on the reservoir, e.g. hierarchical reservoirs (Jaeger, 2007), small-world reservoirs (Deng and Zhang, 2007) and decoupled sub-reservoirs (Xue et al., 2007).

However, there are still serious problems preventing ESN to become a widely accepted tool:

1. There are properties of the reservoir that are poorly understood (Xue et al., 2007),
2. specification of the reservoir and input connections require numerous trials and even luck (Xue et al., 2007),
3. strategies to select different reservoirs for different applications have not been devised (Ozturk et al., 2007),
4. imposing a constraint on spectral radius of the reservoir matrix is a weak tool to properly set the reservoir parameters (Ozturk et al., 2007),
5. the random connectivity and weight structure of the reservoir is unlikely to be optimal and does not give a clear insight into the reservoir dynamics organisation (Ozturk et al., 2007).

Indeed, it is not surprising that part of the scientific community is sceptical about ESNs being used for practical applications (Prokhorov, 2005).

The above problems have been the main motivation of this research.

1.2 Contributions

Typical model construction decisions that an ESN user must make include: setting the reservoir size; setting the sparsity of the reservoir and input connections; setting the ranges for random input and reservoir weights; and setting the reservoir matrix scaling parameter α. The dynamical part of the ESN responsible for input stream coding is treated as a black box which is unsatisfactory from both theoretical and empirical standpoints. First, it is difficult to put a finger on what it actually is in the reservoir's dynamical organisation

that makes ESN so successful. Second, the user is required to tune parameters whose function is not well understood.

Simple reservoir topologies have been proposed as an alternative to the randomised ESN reservoir - e.g. 'feedforward' reservoirs with tape delay connections (Cernansky and Makula, 2005), reservoir with diagonal weight matrix (self-loops) (Fette and Eggert, 2005).

According to the above discussion and current issues, this thesis provides the following contributions:

- It investigates systematically the reservoir construction of Echo State Network (ESN). This thesis proposes two very simple *deterministic* ESN organisation (Simple Cycle reservoir (SCR) in Chapter 3 and Cycle Reservoir with Jumps (CRJ) in Chapter 4). Simple Cycle reservoir (SCR) is sufficient to obtain performances comparable to those of the classical ESN as shown in Section 3.2. While Cycle Reservoir with Jumps (CRJ) significantly outperform the those of the classical ESN as illustrated in Section 4.2.

- It studies and discusses three reservoir characterisations - short-term memory capacity (MC) (Chapter 3 and 4), eigen-spectrum of the reservoir weight matrix (Chapter 4), and Lyapunov Exponent (Chapter 4) with their relation to the ESN performance.

- It designs and utilises an ensemble of ESNs with diverse reservoirs whose collective readout is obtained through Negative Correlation Learning (NCL) of ensemble of Multi-Layer Perceptrons (MLP), where each individual MPL realises the readout from a single ESN (chapter 5).

- It investigates the relation between two quantitative measures characterising short term memory in input driven dynamical systems, namely the short term memory capacity (MC), and the Fisher memory curve (FMC) (chapter 6).

1.3 Thesis Organisation

The remainder of this thesis is organised as follows:

- Chapter 2 gives a broad description of the research context and explains the research questions answered by the thesis.
- Chapter 3 presents a simple deterministically cyclic reservoir that shown performance competitive with standard Echo State Network (ESN).
- Chapter 4 introduces a novel simple deterministic reservoir model, Cycle Reservoir with Jumps (CRJ), with highly constrained weight values, that has superior performance to standard ESN.
- Chapter 5 applies Negative Correlation learning (NCL) to an Ensemble of ESN.
- Chapter 6 investigates the relation between two quantitative measures characterising short term memory in input driven dynamical systems.
- The Conclusions and Future work are drawn in Chapter 7.

1.4 Publications from the Thesis

Some of the material presented in this thesis were published in the following papers:

Journal publications:

1. **Rodan, A. and Tino, P. (2011).** Minimum Complexity Echo State Network, *IEEE Transactions on Neural Networks (TNN)*, 22(1): 131–144. (c) IEEE.

2. **Rodan, A. and Tino, P.(2012).** Simple Deterministically Constructed Cycle Reservoirs with Regular Jumps. *Neural Computation*, 24(7): 1822–1852. (c) MIT Press.

Refereed conference publications:

1. **Rodan, A. and Tino, P. (2010).** Simple Deterministically Constructed Recurrent Neural Networks, *In Intelligent Data Engineering and Automated Learning (IDEAL 2010)*, Lecture Notes in Computer Science, LNCS 6283, Springer-Verlag, pp. 267–274.

2. **Rodan, A. and Tino, P. (2011).** Negatively Correlated Echo State Networks, *In 19th European Symposium on Artificial Neural Networks (ESANN 2011)*, Bruges, Belgium.

3. **Tino, P. and Rodan, A. (2012).** Short Term Memory Quantifications in Input-Driven Linear Dynamical Systems, *In 20th European Symposium on Artificial Neural Networks (ESANN 2012)*, Bruges, Belgium.

Chapter 2

Research Context

This chapter presents in Section 2.1 an overview of Artificial Neural Network covering the most popular learning algorithms. Section 2.2 introduces the Echo Sate Network (ESN), a special type of Recurrent Neural Network (RNN), and one of the simplest, yet effective Reservoir Computing methods. Section 2.3 describes Lyapunov Exponent (LE), one of the characterisation used in the literature to quantify the dynamic properties for a reservoir. Section 2.4 gives an overview of Negative Correlation Learning (NCL), an ensemble learning approach used for Neural Networks. Section 2.5 explains the research questions answered by this work and the motivation behind each of them. Finally, this chapter is summarised in section 2.6.

2.1 Artificial Neural Network

The human brain has the ability to perform multi-tasking. These tasks include controlling the human body temperature, controlling blood pressure, heart rate, breathing, and other tasks that enable human beings to see, hear, and smell and so on. The brain can perform these tasks at a rate that is far less than the rate at which the conventional computer can perform the same tasks (Haykin, 1999). The cerebral cortex of the human brain contains

over 20 billion neurons with each neuron linked with up to 10,000 synaptic connections (Haykin, 1999). These neurons are responsible for transmitting nerve signals to and from the brain. Very little is known about how the brain actually works but there are computer models that try to simulate the same task that the brain carries out. These computer models are called *Artificial Neural Networks*, and the method by which the Neural Network is trained is called a *Learning Algorithm*, which has the duty of training the network and modifying weights in order to obtain a desired response.

The neuron (node) of a neural network is made up of three components:

1. synapse (connection link) which is characterised by its own weight,
2. An adder for summing the input signal, which is weighted by the synapse of the neuron, and
3. An activation function to compute the output of this neuron.

The main Neural Network architectures are Feedforward Neural Network (FFNN) and the Recurrent Neural Network (RNN).

2.1.1 Feedforward Neural Network

The most common and well-known Feedforward Neural Network (FFNN) model is called Multi-Layer Perceptron (MLP). Let a MLP with K input units, N internal (hidden) units, and L output units, where $s = (s_1, s_2..., s_K)^T$, $x = (x_1, x_2..., x_N)^T$, and $y = (y_1, y_2..., y_L)^T$, be the inputs of the input nodes, the outputs of the hidden nodes, and outputs of the output nodes respectively. b_j and b_l are the biases in the input and output layers. A three layer MLP are shown in Figure 2.1.

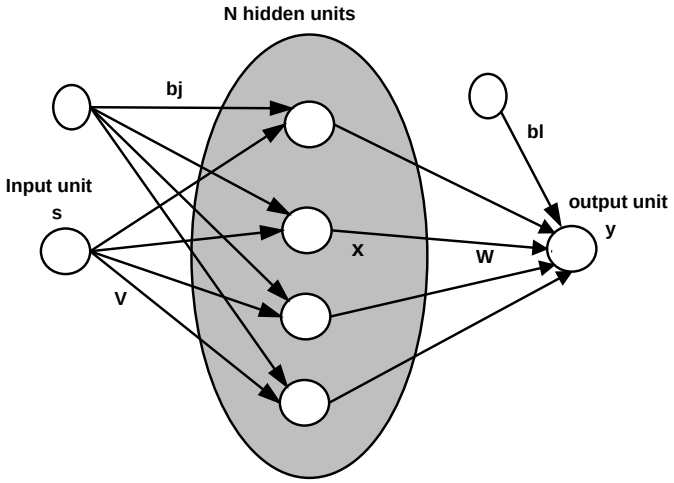

Figure 2.1: An example of the topology of the Multi-layer Perceptron- MLP

In the forward pass the activations are propagated from the input layer to the output layer. The activations of the hidden nodes are the weighted inputs from all the input nodes plus the bias b_j. The activation of the jth hidden node is denoted as net_j, and computed according to:

$$net_j = \sum_{i=1}^{K} V_{ji} s_i + b_j, \tag{2.1}$$

In the hidden layer, the corresponding output of the jth node (e.g. x_j) is usually calculated based on a sigmoid function as follows:

$$x_j = \frac{1}{(1 + e^{-net_j})}, \tag{2.2}$$

The outputs of the hidden layer $(x_1, x_2..., x_N)$ are used as inputs to the output layer. The activation of the output nodes $(y_1, y_2..., y_L)$ is also defined as the weighted inputs from all the hidden nodes plus the bias b_l, where W_{lj} is the connection weight from the

jth hidden node x_j to the lth (linear) output node:

$$y_l = \sum_{j=1}^{N} W_{lj} x_j + b_l, \tag{2.3}$$

The backward pass starts by propagating back the error between the current output y_l and the teacher output $\hat{y}_l$ in order to modify the network weights and the bias values. The MLP network is attempted to minimise the Error (E) via the the classical Backpropagation (BP) training algorithm (Rumelhart et al., 1986), where for each epoch the Error (E) is computed as:

$$E = \sum_{e=1}^{P} \sum_{l=1}^{L} |y_l^e - \hat{y}_l^e|^2, \tag{2.4}$$

where P is the number of patterns.

In MLP all the network weights and bias values are assigned random values initially, and the goal of the training is to find the set of network weights that cause the output of the network to match the teacher values as closely as possible.

MLP has been successfully applied in a number of applications, including regression problems (Brown et al., 2005.), classification problems (Mckay and Abbass, 2001), or time series prediction using simple auto-regressive models (Liu and Yao, 1999), where the output depends only on the current input (*static*). However, there are many tasks that need memory (activities on the context neurons) and their current input depends on the previous inputs to the network (*dynamics*), not only on the current input, so it is difficult to perform these tasks using MLP.

2.1.2 Recurrent Neural Network

Recurrent Neural Network (RNN) (also called Feed-Back Neural Network), is a natural extension of FFNN that contains at least one feedback connection (recurrent or cycle

connection), where an output can be put back into the network to serve as an additional input, which keeps the past information in the unit activation.

Discrete-time Recurrent Neural network (Figure 2.2) is a dynamic neural network with K input units, N internal (hidden) units, and L output units acting in discrete-time steps. Note that there is another type of RNN that works continuously in terms of time steps. The activation of the input, internal, and output units at time step t are denoted by: $s(n) = (s_1(t), ..., s_K(t))^T$, $x(t) = (x_1(t), ..., x_N(t))^T$, and $y(t) = (y_1(t), ..., y_L(t))^T$ respectively. The connections between the input units and the internal units are given by an $N \times K$ weight matrix V, connections between the internal units are collected in an $N \times N$ weight matrix W, and connections from internal units to output units are given in $L \times N$ weight matrix U.

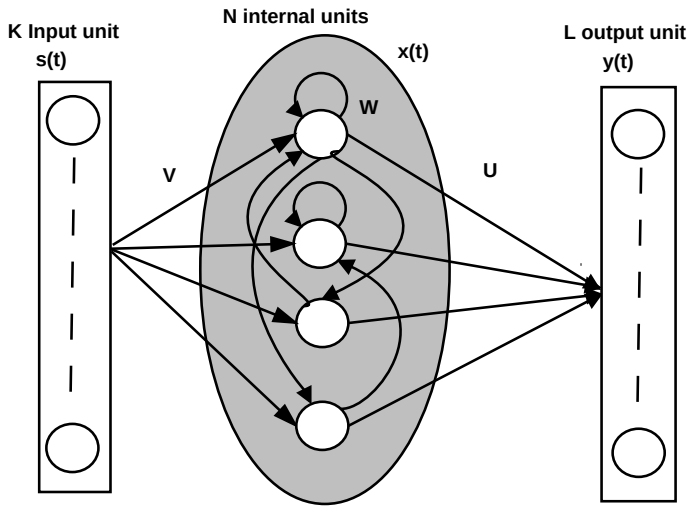

Figure 2.2: An example of Recurrent Neural Network- RNN

For Discrete-time RNN the hidden units are updated according to:

$$x(t+1) = f(s(t+1), x(t)), \tag{2.5}$$

where f is the activation function (typically nonlinear, tanh or some other sigmoidal function); Note that sometimes the output $y(t)$ is also feedback into the hidden layer, in that case we would have:

$$x(t+1) = f(s(t+1), x(t), y(t)), \tag{2.6}$$

The output is computed as:

$$y(t) = g(x(t)), \tag{2.7}$$

where g is the nonlinear output function (typically also tanh or some other sigmoidal function).

Compared with FFNN, RNNs offer more expressive power to approximate nonlinear dynamical systems including regression, classification, learning of context free language, and speech recognition. In RNN all connection weights V, W, and U are adapted using the following popular methods:

- **Backpropagation Through Time (BPTT)**: is an adaptation of the well known Backpropagation learning method (Rumelhart et al., 1986) known from training FeedForward Neural Networks (FFNN), which is the most commonly method used for training Neural Networks. The main idea of BPTT which was proposed first by Werbos (1990), is to 'unfold' the RNN in time, by creating a multilayer feedforward neural network (FFNN) for each time a sequence is processed. Figure 2.3 shows an example of a simple RNN (figure 2.3 left) with its unfolded feedforward version (figure 2.3 right). The training data consists of a number of input-output pairs which is divided into epochs, each epoch has its start time t_{start} and its end time t_{end}. The forward pass of training one epoch consists of updating the multilayer ("unfolded") feedforward network from the first layer $x(t_{start})$ to the last layer $x(t_{end})$. Assume that the error of the current output, the teacher output and the current output

at time t are denoted by: $E(t)$, $y(t)$ and $\hat{y}(t)$ respectively, then the error to be minimised is:

$$E(t_{start}, t_{end}) = \sum_{t=t_{start},...,t_{end}} \|\hat{y}(t) - y(t)\|^2. \tag{2.8}$$

Furthermore, a single backward pass through $t = t_{end}, ..., t_{start}$ is performed to compute the values of the local error gradients which is derived the same way as in standard Backpropagation learning, except that the errors are added in each layer. Finally, the corresponding weights across the layers are updated using the gradient of the error.

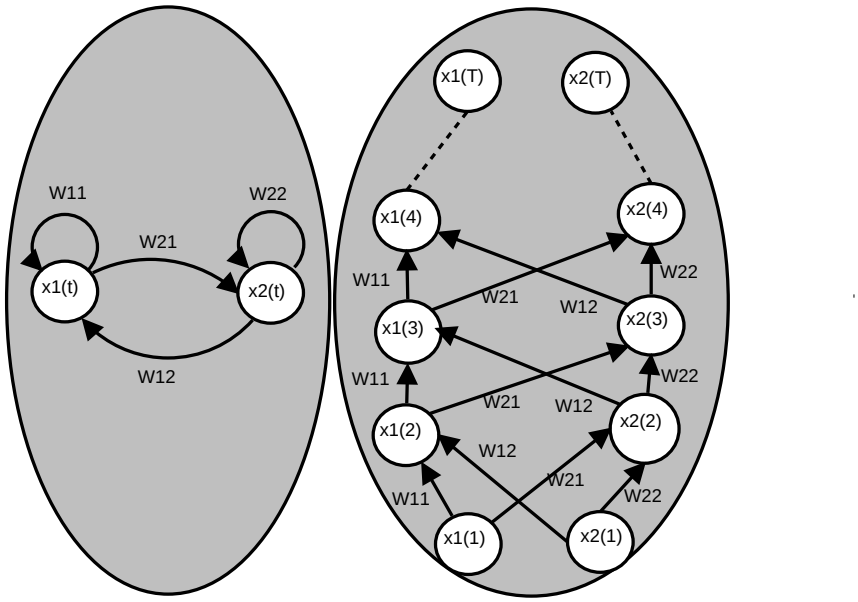

Figure 2.3: An example of a simple RNN (left) and the unfolded feedforward version of the same network (right).

- **Real-time Recurrent Learning (RTRL)**: is an online gradient-descent method described by Williams and Zipser (1989). It computes the exact gradient error at time step t, then it uses this result to compute the forward or the future derivatives at time $t+1$ in a recursive way. Instead of creating a duplicate multi-layer feedforward neural network (FFNN) as in BPTT. RTRL uses a fixed number of parameters to

record training information of the past time, so all the network weights V, W, U are adapted as the new training patterns are introduced.

- **Extended Kalman Filter (EKF)**: is a state estimation technique for nonlinear dynamics and nonlinear measurement equations, derived by linearising the well-known Kalman filter (KF) around the current state estimation. Training neural network by Kalman Filter was first proposed by Singhal and Wu (1989). It was found that EKF-based weight trajectory smoothing training methods gives the best results and outperform the common gradient based algorithms.

2.1.3 Problems of gradient based algorithms

There are still several limitations for using BPTT, like slow convergence, difficulty with local optima, and high computational cost of $O(TN^2)$ for each epoch, which make it not suitable for real-time computations with recurrent neural networks. On the other hand, RTRL also suffers from high computational cost of $O(N^4)$ for each update step for the network weights, so this algorithm is only useful for online training when small network size is sufficient to solve a given problem. It has been also demonstrated very early that gradient based algorithms face the problem of learning dependencies which require long-range memory (Bengio et al., 1994). To (at least partially) overcome this problem long short-term memory (LSTM) networks is proposed in (Gers et al., 1999).
An alternative new paradigm referred to as *reservoir computing (RC)* avoids the problems of gradient based algorithms like slow and difficult progress by designing and training RNN without modifying the transient dynamics of the recurrent network. Echo State Networks (ESNs) (Jaeger, 2001), Liquid State Machines (LSMs) (Maass et al., 2002) and the back-propagation decorrelation neural network (BPDC) (Steil, 2004) are the most popular examples of this new paradigm.

In this work we concentrate on Echo State Networks, one of the simplest, yet effective

form of reservoir computing.

2.2 Echo State Network (ESN)

An echo state network is a recurrent discrete-time neural network with K input units, N internal (reservoir) units, and L output units. The activation of the input, internal, and output units at time step t are denoted by: $s(t) = (s_1(t), ..., s_K(t))^T$, $x(t) = (x_1(t), ..., x_N(t))^T$, and $y(t) = (y_1(t), ..., y_L(t))^T$ respectively. The connections between the input units and the internal units are given by an $N \times K$ weight matrix V, connections between the internal units are collected in an $N \times N$ weight matrix W, and connections from internal units to output units are given in $L \times N$ weight matrix U.

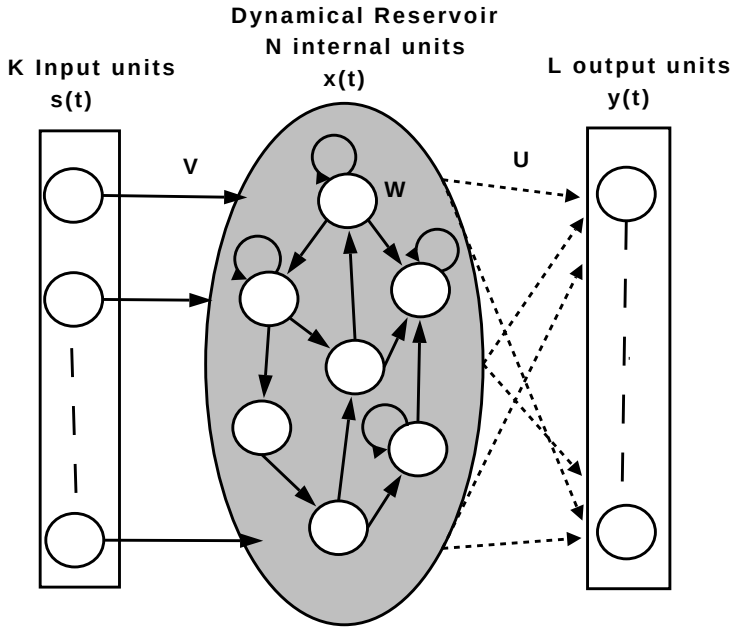

Figure 2.4: Echo state network (ESN) Architecture

The internal units are updated according to:

$$x(t+1) = f(Vs(t+1) + Wx(t) + z(t+1)), \tag{2.9}$$

where f is the reservoir activation function (typically tanh or some other sigmoidal function); $z(t+1)$ is an optional uniform i.i.d. noise. In this ESN model, there are no feedback connections from the output to the reservoir and no direct connections from the input to the output.

The linear readout is computed as:

$$y(t+1) = Ux(t+1). \tag{2.10}$$

The reservoir activation vector x is extended with a fixed element accounting for the bias term. Elements of W and V are fixed prior to training with random values drawn from a uniform distribution over a (typically) symmetric interval, where only the output connection weights U are adapted using any linear regression method.

In order for ESN to "work", the reservoir with weights W should have the *"Echo State Property"* (ESP). ESP says that the reservoir state is an *"echo"* of the entire input history and the reservoir will wash out any information from initial conditions. To account for ESP, the eigenvalues of W should lie inside the unit circle by scaling the reservoir connection weights W as $W \leftarrow \alpha W/|\lambda_{max}|$, where $|\lambda_{max}|$ is the spectral radius, which is the largest among the absolute values of the eigenvalues of W and $0 < \alpha < 1$ is a scaling parameter.

ESN memoryless readout can be trained both offline (Batch) and online by minimising a given loss function. In most cases we evaluate the model performance via Normalised Mean Square Error (NMSE):

$$NMSE = \frac{\langle \|\hat{y}(t) - y(t)\|^2 \rangle}{\langle \|y(t) - \langle y(t) \rangle\|^2 \rangle}, \tag{2.11}$$

where $\hat{y}(t)$ is the readout output, $y(t)$ is the desired output (target), $\|.\|$ denotes the Euclidean norm and $\langle \cdot \rangle$ denotes the empirical mean.

2.2.1 Offline (Batch) Training

In the *offline (batch) training* mode one first runs the network on the training set, and subsequently computes the output weights that minimise the NMSE. In summary, the following steps are performed:

1. Initialise W with a scaling parameter $\alpha < 1$ and run the ESN on the training set.
2. Dismiss data from initial *washout* period and collect the remaining network states $x(t)$ row-wise into a matrix x, where in case of direct input-output connections, the matrix x collects inputs $s(t)$ as well.
3. The target values from the training set are collected in a vector y.
4. The output unit weights are computed using one of the following four methods:

- Singular value Decomposition (SVD): SVD of an $M \times N$ matrix x is of the form $x = P.S.Q^T$, where T denotes transpose operation, P and Q are $M \times M$ and $N \times N$ orthonormal matrices respectively, and S is an $M \times N$ diagonal matrix containing singular values $\delta_{11} \geq \delta_{22} \geq ... \geq \delta_{NN} \geq 0$. Output weights U are found by solving $x.U = y$.
- Pseudoinverse Solution: The output weights U are computed by multiplying the pseudoinverse of x with y and transposing the result, that is, $U = (x^{\dagger}.y)^T$.
- Wiener-Hopf Solution: The output weights U are computed by $U = M^{-1}.D$ where $M = x^T.x$ is the correlation matrix of the reservoir states and $D = x^T.y$ is the cross-correlation matrix of the states vs. the target (desired) outputs.
- Ridge Regression: The Output weights U are computed as

$$U = (x^T x + \lambda^2 I)^{-1}\, x^T y, \tag{2.12}$$

where I is the identity matrix and $\lambda > 0$ is a regularisation factor determined on a hold-out validation set .

SVD, Pseudoinverse and Wiener-Hopf methods are, in principle, similar and equivalent to each other, if x is full rank (number of reservoir units). If this is not the case, i.e. the matrix M of the Wiener-Hopf solution is ill-conditioned, the Pseudoinverse and SVD is numerically stable, while Wiener-Hopf solution is not.

2.2.2 Online Training

Standard recursive algorithms, such as Recursive Least Squares (RLS), for NMSE minimisation can be used in *online readout training*. In RLS, after the initial washout period the output weights U are recursively updated at every time step t:

$$k(t) = \frac{\phi(t-1)\ x(t)}{x^T(t)\ \phi(t-1)\ x(t) + \gamma} \tag{2.13}$$

$$\phi(t) = \gamma^{-1}(\phi(t-1) - k(t)\ x^T(t)\ \phi(t-1)) \tag{2.14}$$

$$U(t) = U(t-1) + k(t)\ [y(t) - \hat{y}(t)] \tag{2.15}$$

where k stands for the innovation vector; y and $\hat{y}$ correspond to the desired and calculated (readout) output unit activities; ϕ is the error covariance matrix initialised with large diagonal values. 'Forgetting parameter' $0 < \gamma < 1$ is usually set to a value close to 1.0. In this work γ is set on a hold-out validation set.

2.2.3 Short Term Memory Capacity of ESN

Jaeger (2002a) quantified the inherent capacity of recurrent network architectures to represent past events through a measure correlating the past events in an i.i.d. input stream

with the network output. In particular, assume that the network is driven by a univariate stationary input signal $s(t)$. For a given delay k, we consider the network with optimal parameters for the task of outputting $s(t-k)$ after seeing the input stream $...s(t-1)s(t)$ up to time t. The goodness of fit is measured in terms of the squared correlation coefficient between the desired output (input signal delayed by k time steps) and the observed network output $y(t)$:

$$MC_k = \frac{Cov^2(s(t-k), y(t))}{Var(s(t))\ Var(y(t))}, \tag{2.16}$$

where Cov denotes the covariance and Var the variance operators. The short term memory (STM) capacity is then given by (Jaeger, 2002a):

$$MC = \sum_{k=1}^{\infty} MC_k. \tag{2.17}$$

Jaeger (2002a) proved that for *any* recurrent neural network with N recurrent neurons, under the assumption of i.i.d. input stream, the STM capacity cannot exceed N, where N is the number of reservoir units.

2.3 Lyapunov Exponent

The 'edge of chaos' is a regime of a dynamical system so that it operates at the boundary between the 'chaos' and 'order'. In this regime, the dynamical system can demonstrate a high computational power (Bertschinger and Natschlager, 2004; Legenstein and Maass, 2005), where the effect of the input on the reservoir states does not die quickly (Legenstein and Maass, 2005). However, this does not universally imply that such reservoirs are optimal (Legenstein and Maass, 2007). The 'edge of chaos' can be numerically calculated for biological reservoirs by computing the pseudo-Lyapunov Exponent (LE) (Verstraeten et al., 2007). LE is one of the characterisation used in the literature to quantify the dynamic properties for a reservoir and it can be determined by computing the Jacobian

matrix $J_f(x)$ of the reservoir derivative states x (Verstraeten et al., 2010):

$$J_f(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) ... \frac{\partial f_1}{\partial x_N}(x) \\ \frac{\partial f_N}{\partial x_1}(x) ... \frac{\partial f_N}{\partial x_N}(x) \end{pmatrix}, \tag{2.18}$$

where f is the dynamic function (see eq. 2.9), ∂ is the partial derivative, and $x = [x_1 x_2 ... x_N]$ is the states for all the reservoir units. $J_f(x)$ can be simplified as (Verstraeten et al., 2010):

$$J_f(x) = diag[1 - x_1^2(t), 1 - x_2^2(t), 1 - x_N^2(t)]W, \tag{2.19}$$

where $diag[]$ presents the diagonal matrix with the diagonal values. From this, the kth LE λ_k can be approximated as $log(\prod_{t=1}^{M}(r_k)^{\frac{1}{t}})$, where M is the number of time steps, and r_k is the kth eigenvalue spectrum of the Jacobian matrix $J_f(x)$ (Verstraeten et al., 2010). A note of caution is needed here: The largest exponents thus collected are then used to produce an estimate of the average exponential divergence rate of nearby trajectories along the input-driven reservoir trajectory. Even though for input-driven systems this is only a heuristic measure, where deep results of autonomous systems theory e.g. linking positive Lyapunov exponents to topological entropy (Pesin Theorem) no longer apply, nor do apply traditional notions of 'chaos' and 'order' developed in the context of autonomous systems, it nevertheless proved useful in suggesting the 'optimal' reservoir configuration across several tasks (Verstraeten et al., 2007).

2.4 Negative Correlation Learning (NCL)

It has been extensively shown that ensemble learning can offer a number of advantages over a single learning machine (e.g. neural network) training. It has a potential to e.g. improve generalisation and decrease the dependency on training data (Brown and Yao, 2001). One of the key elements for building ensemble models is the "diversity" among individual ensemble members. Negative correlation learning (NCL) (Liu and Yao, 1999)

is an ensemble learning technique that encourages diversity among ensemble members through their negative correlation, while keeping the training error small. It has been successfully applied to training Multi Layer Perceptron (MLP) ensembles in a number of applications, including regression problems (Brown et al., 2005.), classification problems (Mckay and Abbass, 2001), or time series prediction using simple auto-regressive models (Liu and Yao, 1999).

In NCL, all the individual networks are trained simultaneously and interactively through the correlation penalty terms in their error functions. The procedure has the following form: Given a set of M networks and a training input set s, the ensemble output $F(t)$ is calculated as a flat average over all ensemble members $F_i(t)$,

$$F(t) = \frac{1}{M}\sum_{i=1}^{M}(F_i(t)). \tag{2.20}$$

In NCL the penalised error functional to be minimised reads:

$$E_i = \frac{1}{2}(F_i(t) - y(t))^2 + \lambda p_i(t), \tag{2.21}$$

where

$$p_i(t) = (F_i(t) - F(t))\sum_{i\neq j}(F_j(t) - F(t)), \tag{2.22}$$

and $\lambda > 0$ is an adjustable strength parameter for the negative correlation enforcing penalty term p_i. It can be shown that

$$E_i = \frac{1}{2}(F_i(t) - y(t))^2 - \lambda(F_i(t) - F(t))^2. \tag{2.23}$$

Note that when $\lambda = 0$, we obtain a standard de-coupled training of individual ensemble members. Standard gradient-based approaches, which have been described in section 2.1.2, can be used to minimise E by updating the parameters of each individual ensemble

member.

2.5 Research Questions

This Section explains the research questions answered by this work and the motivation behind each of them. More detailed motivations for the way to tackle/solve each question/problem are given in each one of the Sections related to the proposed solutions.

- **What is the minimal complexity of the reservoir topology and parametrisation so that performance levels comparable to those of standard reservoir computing models, such as ESN, can be recovered?, and What degree of randomness (if any) is needed to construct competitive reservoirs?**

Echo State Network (ESN) is a recurrent neural network (RNN) with a non-trainable fixed sparse recurrent layer (reservoir), where the connection weights in the ESN reservoir, as well as the input weights are randomly generated. So, it is important to investigate the reservoir construction of Echo State Network (ESN). In particular, Section 3.2 shows that very simple ESN organisation is sufficient to obtain performances comparable to those of the classical ESN, where for a variety of tasks it is sufficient to consider:

1. a simple fixed non-random reservoir topology with full connectivity from inputs to the reservoir
2. a single fixed absolute weight value r for all reservoir connections and
3. a single weight value v for input connections, with (deterministically generated) aperiodic pattern of input signs.

The results shown in Section 3.2.3 indicate that comparable performances of Simple Cycle Reservoir (SCR) topology can be obtained without any stochasticity in the input

weight generation by consistent use of the same sign generating algorithm across a variety of data sets.

- **If simple competitive reservoirs constructed in a completely deterministic manner exist, how do they compare in terms of memory capacity with established models such as recurrent neural networks? and, What is the memory capacity of such simplified reservoirs?**

Jaeger (2002a) proved that the inherent capacity (Short term memory capacity (STM)) for *any* recurrent neural network with N recurrent neurons, under the assumption of i.i.d. input stream, cannot exceed N, where N is the number of reservoir units.
We prove in Section 3.3 (under the assumption of zero-mean i.i.d. input stream) that the Short term memory (STM) capacity of linear Simple Cycle Reservoir (SCR) architecture with N reservoir units can be made arbitrarily close to N.

- **Can the extending of the Simple Cycle Reservoir (SCR) introduced in Section 3.1 with a regular structure of shortcuts (Jumps) by keeping the reservoir construction simple and deterministic, significantly outperform the standard randomised ESN?**

In chapter 3 we argue that randomisation and trail-and-error construction of reservoirs may not be necessary. Very simple, cyclic, deterministically generated reservoirs are shown to yield performance competitive with standard ESN on a variety of data sets of different origin and memory structure.

In particular, Section 4.1 introduces a novel simple deterministic reservoir model, Cycle Reservoir with Jumps (CRJ), with highly constrained weight values, that has superior performance to standard ESN on a variety of temporal tasks of different origin and characteristics. It seems that the long-held belief that the randomised generation of reservoirs is somehow crucial for allowing a wide variety of dynamical features in the

reservoir may not be true.

- **Are reservoir characterisations, such as memory capacity, eigenvalue distribution of the reservoir matrix or pseudo-Lyapunov exponent of the input-driven reservoir dynamics related to ESN model performance?**

In Section 3.3 (under the assumption of zero-mean i.i.d. input stream) the MC of linear SCR architecture with N reservoir units can be made arbitrarily close to N. In particular, $MC = N - (1 - r^{2N})$, where $r \in (0, 1)$ is the single weight value for all connections in the cyclic reservoir. In Section 4.4.2 we present a new framework for determining short term memory capacity of linear reservoir models to a high degree of precision. Using the framework we study the effect of shortcut (jumps) connections in the CRJ reservoir topology on its memory capacity. Due to cross-talk effects introduced by the jumps in CRJ, the MC contributions start to rapidly decrease earlier than in the case of SCR, but unlike in SCR, the decrease in MC_k in CRJ is gradual, enabling the reservoir to keep more information about some of the later inputs.

Furthermore, it has been also been suggested that a uniform coverage of the unit disk by such eigenvalues can lead to superior model performances. We show in Section 4.4.1 that this is not necessarily so. Despite having highly constrained eigenvalue distribution the CRJ consistently outperforms ESN with much more uniform eigenvalue coverage of the unit disk. Moreover, unlike in the case of ESN, pseudo-Lyapunov exponents of the selected 'optimal' CRJ models are consistently negative (see Section 4.4.3).

- **Can the use of Negative Correlation Learning (NCL) for state space modelling such as recurrent neural network (RNN) achieve better generalisation performance?**

There have been studies of simple ESN ensembles (Schwenker and Labib, 2009), or Multi-Layer Perceptron (MLP) readouts (Babinec and Pospichal, 2006; Bush and Anderson, 2005), but to the best of our knowledge, this is the first study employing a NCL style training in ensembles of state space models, such as ESNs, where in comparison with both single ESN and flat ensembles of ESNs, Section 5.3 shows that NCL based ESN ensembles achieve better generalisation performance. The last research question answered by the thesis is:

- **Is there any relationship between two of the main well known measures used to characterise short term memory in input driven dynamical systems, namely the short term memory capacity spectrum and the Fisher memory curve?**

  In Section 6.2, we show that under some assumptions, the two measures can be interpreted as squared 'Mahalanobis' norms of images of the input vector under the system's dynamics and that $MC_k > \epsilon\ J(k)$, for all $k > 0$. Even though MC_k and $J(k)$ map the memory structure of the system under investigation from two quite different perspectives, they can be closely related.

2.6 Chapter Summary

We have introduced the research context for this work, where in Section 2.1 we had an overview about Artificial Neural Network covering some of the most important learning algorithms. In Section 2.2 we gave a detailed description of Echo State Network (ESN) which is a special type of RNN, and one of the simplest, yet most effective reservoir Computing methods, that we will use as a baseline model for our work throughout the thesis. Section 2.3 described Lyapunov Exponent (LE), one of the characterisation used in the literature to quantify the dynamic properties for a reservoir. Section 2.4 presented an overview about Negative Correlation Learning (NCL), which will be used to design

an ensemble of ESNs with diverse reservoirs whose collective readout is obtained through NCL of ensemble of Multi-Layer Perceptrons (MLP). Finally, Section 2.5 presented the research questions answered by the thesis.

Chapter 3

Minimum Complexity Echo State Network

In this chapter we would like to systematically investigate the reservoir construction of Echo State Network (ESN); namely we show that in fact a very simple ESN organisation is sufficient to obtain performances comparable to those of the classical ESN. We argue that for a variety of tasks it is sufficient to consider:

1. a simple fixed non-random reservoir topology with full connectivity from inputs to the reservoir ,
2. a single fixed absolute weight value r for all reservoir connections and
3. a single weight value v for input connections, with *deterministically* generated "pseudo-random" aperiodic pattern of input signs.

The rest of the chapter is organised as follows. Section 3.1 presents our simplified reservoir topologies. Experimental results are presented in Section 3.2. We analyse both theoretically and empirically the short term memory capacity (MC) of our simple reservoir in Section 3.3. Finally, this chapter is summarised in Section 3.4.

3.1 Simple Echo state network reservoirs

To simplify the reservoir construction, we propose several easily structured topology templates and we compare them to those of the classical ESN. We consider both *linear reservoirs* that consist of neurons with identity activation function, as well as *non-linear reservoirs* consisting of neurons with the commonly used tangent hyperbolic (tanh) activation function. Linear reservoirs are fast to simulate but often lead to inferior performance when compared to non-linear ones (Verstraeten et al., 2007).

3.1.1 Reservoir Topology

We consider the following three reservoir templates (model classes) with fixed topologies Figure. 3.1 :

- *Delay Line Reservoir (DLR)* - composed of units organised in a line. Only elements on the lower sub-diagonal of the reservoir matrix W have non-zero values $W_{i+1,i} = r$ for $i = 1...N-1$, where r is the weight of all the feedforward connections.
- *DLR with feedback connections (DLRB)* - the same structure as DLR but each reservoir unit is also connected to the preceding neuron. Nonzero elements of W are on the lower $W_{i+1,i} = r$ and upper $W_{i,i+1} = b$ sub-diagonals, where b is the weight of all the feedback connections.
- *Simple Cycle Reservoir (SCR)* - units organised in a cycle. Nonzero elements of W are on the lower sub-diagonal $W_{i+1,i} = r$ and at the upper-right corner $W_{1,N} = r$.

3.1.2 Input Weight Structure

The input layer is fully connected to the reservoir and all input connections have the same absolute weight value $v > 0$; the sign of each input weight is determined randomly by

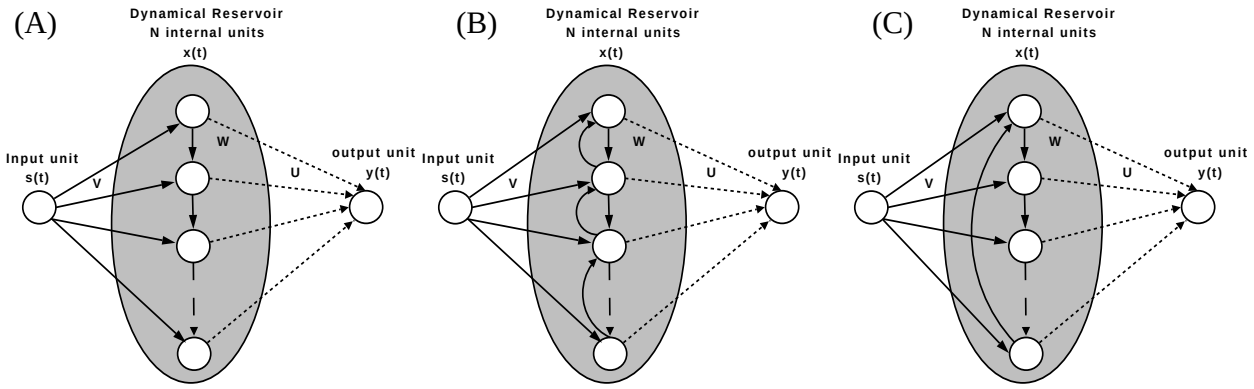

Figure 3.1: (A) Delay Line Reservoir (DLR). (B) Delay Line Reservoir with feedback connections (DLRB). (C) Simple Cycle Reservoir (SCR).

a random draw from Bernoulli distribution of mean 1/2 (unbiased coin). The value v is chosen on the validation set.

3.2 Experiments

3.2.1 Datasets

We use a range of timeseries covering a wide spectrum of memory structure and widely used in the ESN literature (Schrauwen et al., 2008b; Cernansky and Tino, 2008; Jaeger, 2001, 2002a, 2003; Jaeger and Hass, 2004; Verstraeten et al., 2007; Steil, 2007). For each data set, we denote the length of the training, validation and test sequences by L_{trn}, L_{val} and L_{tst}, respectively. The first L_{wash} values from training, validation and test sequences are used as the initial washout period.

NARMA System

The Non-linear Auto-Regressive Moving Average ($NARMA$) system is a discrete time system. This system was introduced in (Atiya and Parlos, 2000). The current output depends on both the input and the previous output. In general, modelling this system is

difficult, due to the non-linearity and possibly long memory.

- *fixed order NARMA time series: NARMA* systems of order $O = 10, 20$ given by equations 3.1, and 3.2, respectively.

$$y(t+1) = 0.3y(t) + 0.05y(t)\sum_{i=0}^{9} y(t-i) + 1.5s(t-9)s(t) + 0.1, \tag{3.1}$$

$$\begin{aligned} y(t+1) = &\tanh(0.3y(t) + 0.05y(t)\sum_{i=0}^{19} y(t-i) \\ &+ 1.5s(t-19)s(t) + 0.01), \end{aligned} \tag{3.2}$$

where $y(t)$ is the system output at time t, $s(t)$ is the system input at time t (an i.i.d stream of values generated uniformly from an interval $[0, 0.5]$) (Atiya and Parlos, 2000; Jaeger, 2003).

-*random 10th order NARMA time series:* This system is generated by:

$$y(t+1) = \tanh(\alpha y(t) + \beta y(t)\sum_{i=0}^{9} y(t-i) + \gamma s(t-9)s(t) + \varphi), \tag{3.3}$$

where α, β, γ and φ are assigned random values taken from $\pm 50\%$ interval around their original values in eq. (3.1) (Jaeger, 2003). Since the system is not stable, we used a non-linear saturation function *tanh* (Jaeger, 2003). The input $s(t)$ and target data $y(t)$ are shifted by -0.5 and scaled by 2 as in (Schrauwen et al., 2008b). The networks were trained on system identification task to output $y(t)$ based on $s(t)$, with $L_{trn} = 2000$, $L_{val} = 3000$, $L_{tst} = 3000$ and $L_{wash} = 200$.

Laser Dataset

The Santa Fe Laser dataset (Jaeger et al., 2007a) is a cross-cut through periodic to chaotic intensity pulsations of a real laser. A fragment of the laser dataset is presented in figure 3.2. The task is to predict the next laser activation $y(t+1)$, given the values up to time

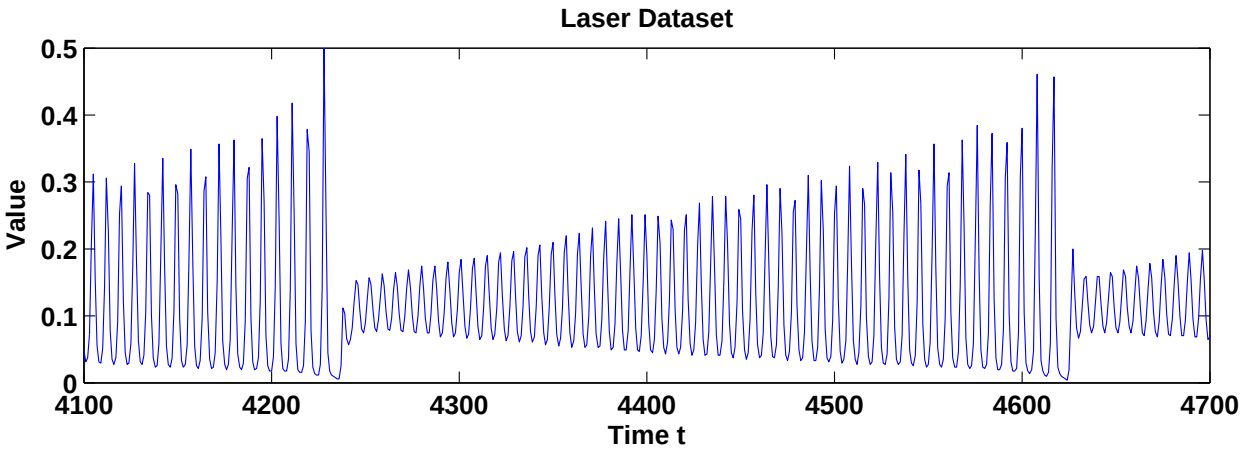

Figure 3.2: A fragment of the laser dataset.

t; $L_{trn} = 2000$, $L_{val} = 3000$, $L_{tst} = 3000$ and $L_{wash} = 200$.

Hénon Map

Hénon Map dataset (Henon, 1976) is generated by:

$$y(t) = 1 - 1.4y(t-1)^2 + 0.3y(t-2) + z(t), \tag{3.4}$$

where $y(t)$ is the system output at time t, $z(t)$ is a normal white noise with standard deviation of 0.05 (Slutzky et al., 2003). We used $L_{trn} = 2000$, $L_{val} = 3000$, $L_{tst} = 3000$ and $L_{wash} = 200$. The dataset is shifted by -0.5 and scaled by 2. Again, the task is to predict the next value $y(t+1)$, given the values up to time t.

Non-linear Communication Channel

The dataset was created as follows (Jaeger and Hass, 2004): first, an i.i.d. sequence $d(t)$ of symbols transmitted through the channel is generated by randomly choosing values from $\{-3, -1, 1, 3\}$ (uniform distribution). Then, $d(t)$ values are used to form a sequence

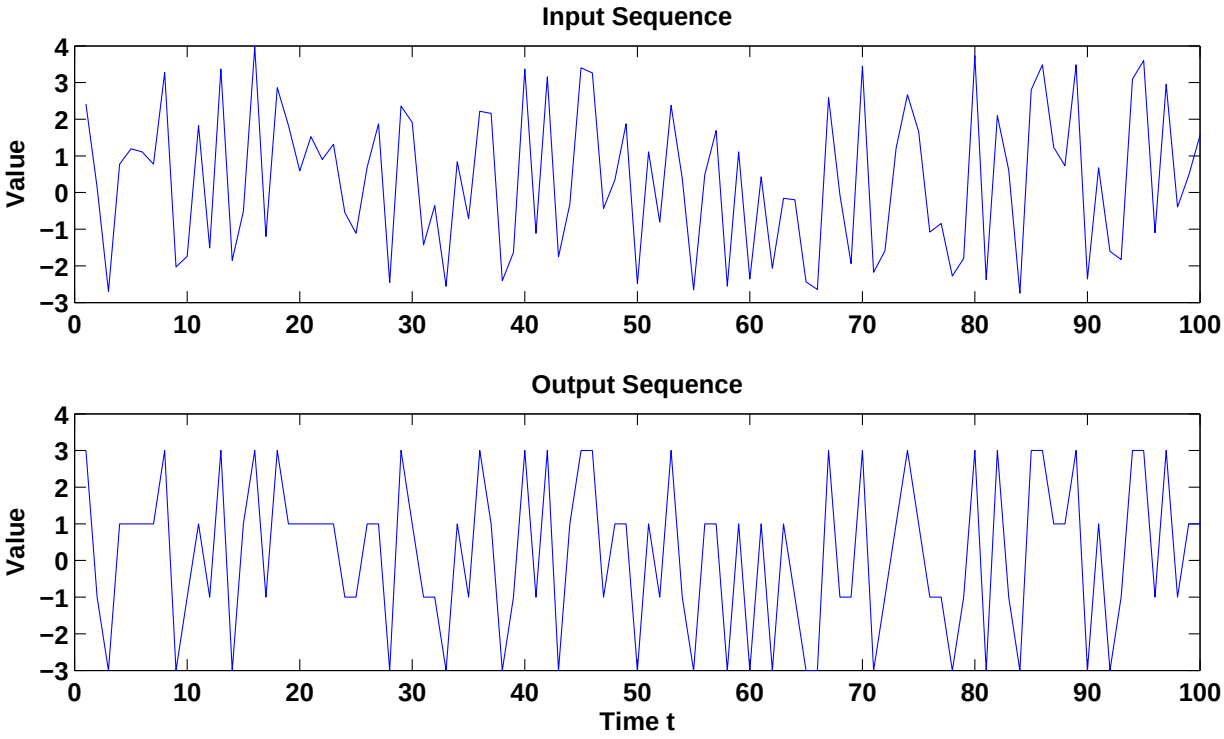

Figure 3.3: A sample of the input $s(t)$ and output $d(t)$ signals of the non-linear communication channel dataset.

$q(t)$ through a linear filter

$$\begin{aligned} q(t) =& 0.08d(t+2) - 0.12d(t+1) + d(t) + 0.18d(t-1) \\ & - 0.1d(t-2) + 0.09d(t-3) - 0.05d(t-4) \\ & + 0.04d(t-5) + 0.03d(t-6) + 0.01d(t-7). \end{aligned} \tag{3.5}$$

Finally, a non-linear transformation is applied to $q(n)$ to produce the signal $s(t)$:

$$s(t) = q(t) + 0.0036q(t)^2 - 0.11q(t)^3. \tag{3.6}$$

A sample of the input $s(t)$ and output $d(t)$ signals are presented in figure 3.3. Following (Jaeger and Hass, 2004), the input $s(t)$ signal was shifted +30. The task is to output $d(t-2)$ when $s(t)$ is presented at the network input. $L_{trn} = 2000$, $L_{val} = 3000$,

$L_{tst} = 3000$ and $L_{wash} = 200$.

IPIX Radar

The sequence used by Xue et al. (2007) contains 2000 values with $L_{trn} = 800$, $L_{val} = 500$, $L_{tst} = 700$ and $L_{wash} = 100$. The target signal is the sea clutter data (the radar backscatter from an ocean surface). The task was to predict $y(t+1)$ and $y(t+5)$ (1 and 5 step ahead prediction) when $y(t)$ is presented at the network input.

Sunspot series

The dataset contains 3100 sunspots numbers from Jan 1749 to April 2007, where $L_{trn} = 1600$, $L_{val} = 500$, $L_{tst} = 1000$ and $L_{wash} = 100$. The task was to predict the next value $y(t+1)$ based on the history of y up to time t.

Non-linear System with Observational Noise

This system was studied in (Gordon et al., 1993.) in the context of Bayesian Sequential State estimation. The data is generated by:

$$s(t) = 0.5s(t-1) + 25\frac{s(t-1)}{1+s^2(t-1)} + 8\cos(1.2s(t-1)) + w(t), \tag{3.7}$$

$$y(t) = \frac{s^2(t)}{20} + v(t), \tag{3.8}$$

where the initial condition is $s(0) = 0.1$; $w(t)$ and $v(t)$ are zero-mean Gaussian noise terms with variances taken from $\{1, 10\}$, i.e. $(\sigma_w^2, \sigma_v^2) \in \{1, 10\}^2$. $L_{trn} = 2000$, $L_{val} = 3000$, $L_{tst} = 3000$ and $L_{wash} = 200$. The task was to predict the value $y(t+5)$, given the values from $t-5$ up to time t presented at the network input.

Isolated Digits

This dataset is a subset of the TI46 dataset which contains 500 spoken *Isolated Digits* (zero to nine), where each digit is spoken 10 times by 5 female speakers. These 500 digits are randomly split into training ($N_{trn} = 250$) and test ($N_{tst} = 250$) sets. Because of the limited amount of data, model selection was performed using 10-fold cross-validation on the training set. The Lyon Passive Ear model (Lyon, 1982) is used to convert the spoken digits into 86 frequency channels. Following the ESN literature using this dataset, the model performance will be evaluated using the Word Error Rate (WER), which is the number of incorrect classified words divided by the total number of presented words. The 10 output classifiers are trained to output 1 if the corresponding digit is uttered and -1 otherwise. Following (Schrauwen et al., 2007a) the temporal mean over complete sample of each spoken digit is calculated for the 10 output classifiers. The Winner-Take-All (WTA) methodology is then applied to estimate the spoken digit's identity. We use this data set to demonstrate the modelling capabilities of different reservoir models on high-dimensional (86 input channels) time series.

3.2.2 Training

We trained a classical ESN, as well as SCR, DLR, and DLRB models (with linear and tanh reservoir nodes) on the time series described above with the NMSE to be minimised. For each model we calculate the average NMSE (in case of *Isolated Digits* dataset, word error Rate (WER) was used) over 10 simulation runs. The model fitting was done using both offline (Batch) and online training.

- For offline training we used ridge regression, where the regularisation factor λ was tuned per reservoir and per dataset on the validation set. We also tried other forms of offline readout training, such as wiener-hopf methodology (e.g. (Ozturk et al., 2007)), pseudoinverse solution (e.g (Jaeger, 2001)), and singular value decomposition

(e.g. (Cernansky and Tino, 2008)), which we described in detail in section 2.2.1. Ridge regression led to the best results.

- For online training we used RLS with forgetting factor of $\gamma = 0.9999995$ (Jaeger, 2003), and we add uniform noise $z(t)$ to the updated internal unit activations (Jaeger, 2003), where the noise level (a form of regularisation) was optimised per reservoir and per dataset using the validation set.

Our experiments are organised along five degrees of freedom:

1. Reservoir topology.
2. Reservoir activation function.
3. Input weight structure.
4. Readout learning.
5. Reservoir size.

3.2.3 Results

For each data set and each model class (ESN, DLR, DLRB, SCR) we picked on the validation set a model representative to be evaluated on the test set. Ten randomisations of each model representative were then tested on the test set.

- For the DLR, DLRB and SCR architectures the model representatives are defined by the method of readout learning, the input weight value v and the reservoir weight r (for DLRB network we also need to specify the value b of the feedback connection). The randomisation was performed solely by randomly generating the signs for individual input weights, the reservoir itself was intact. Strictly speaking we randomly generated the signs for input weights and input biases. However, as

usual in the neural network literature, the bias terms can be represented as input weights from a constant input +1.

- For the ESN architecture, the model representative is specified by readout learning, input weight scaling, reservoir sparsity and spectral radius of the weight matrix. Input weights are (as usual) generated randomly from a uniform distribution over an interval $[-a, a]$.

For each model setting (e.g. for ESN - readout learning, input weight scaling, reservoir sparsity and spectral radius), we generate 10 randomised models and calculate their average validation set performance. The best performing model setting on the validation set is then used to generate another set of 10 randomised models that are fitted on the training set and subsequently tested on the test set. More details about the experiments, such as the chosen readout learning method, input and reservoir weights, spectral radius of the reservoir weight matrix ect. can be found in Appendix A Tables A.1 and A.2.

Figures 3.4, 3.5, 3.6 and 3.7(A) show the average test set NMSE (across ten randomisations) achieved by the selected model representatives. Figure 3.4 presents results for the four model classes using non-linear reservoir on the *laser*, *Hénon Map* and *Non-linear Communication Channel* datasets. On those time series, the test NMSE for linear reservoirs were of an order of magnitude worse than the NMSE achieved by the non-linear ones. While the ESN architecture slightly outperforms the simplified reservoirs on the *laser* and *Hénon Map* time series, for the *Non-linear Communication Channel* the best performing architecture is the simple delay line network (DLR). The SCR reservoir is consistently the second-best performing architecture. The differences between NMSE of ESN and SCR on the *Non-linear Communication Channel* for all reservoir sizes ($N = 50, 100, 150, 200$) are statistically significant at 95% significance level (p values were smaller than 0.05). For reservoir sizes $N = 100$ and $N = 200$, the significance of the differences was high ($p \approx 0.0006$ and $p \approx 0.00007$, respectively). Note that the *Non-linear Communication Channel* can be modelled rather well with a simple Markovian delay line reservoir and no

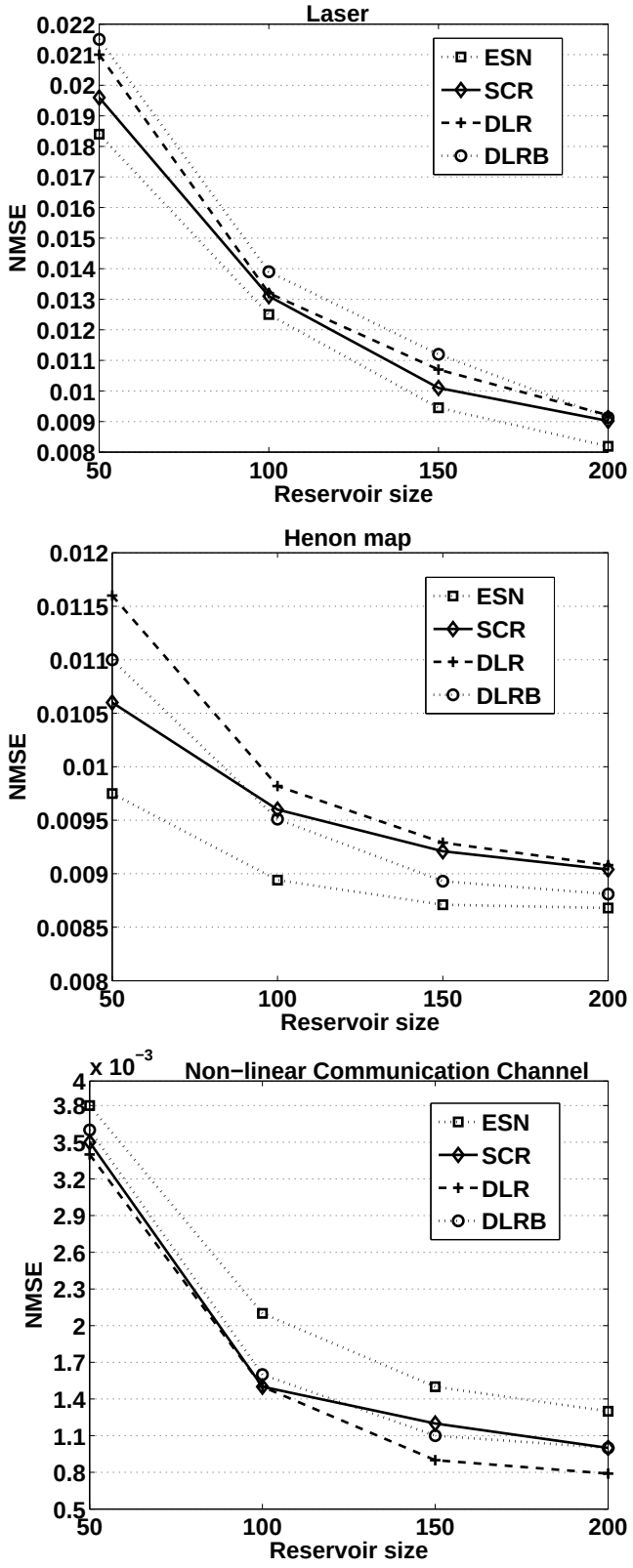

Figure 3.4: Test set performance of ESN, SCR, DLR, and DLRB topologies with *tanh* transfer function on the *laser*, *Hénon Map*, and *Non-linear Communication Channel* datasets.

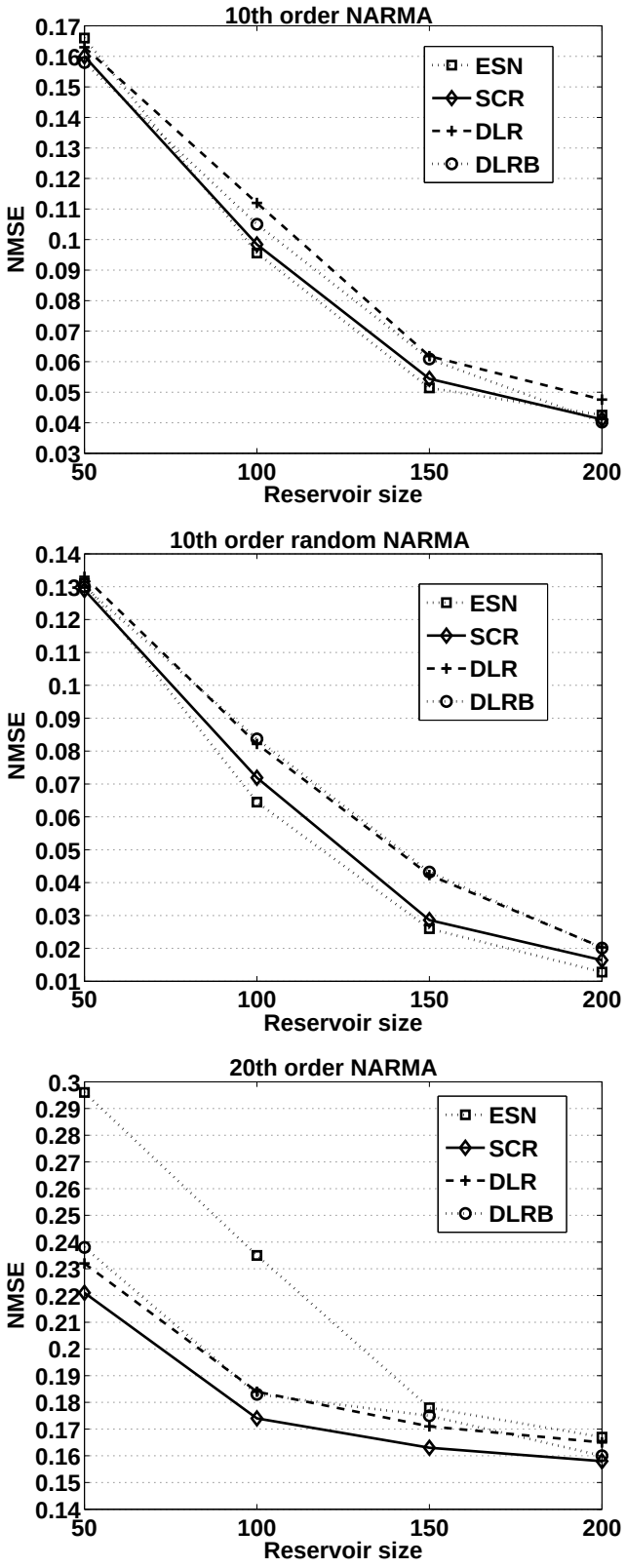

Figure 3.5: Test set performance of ESN, SCR, DLR, and DLRB topologies with *tanh* transfer function on *10th-order, random 10th-order* and *20th-order NARMA* datasets.

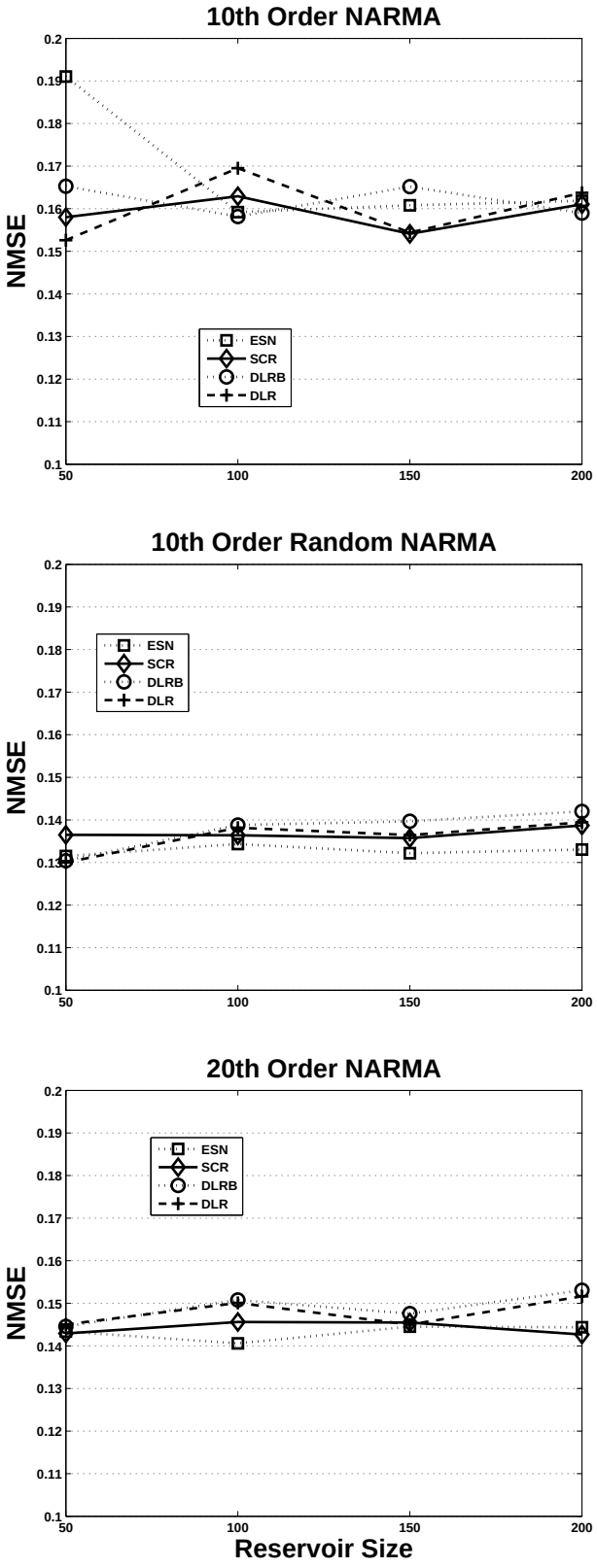

Figure 3.6: Test set performance of ESN, SCR, DLR, and DLRB topologies with *linear* transfer function on *10th-order, random 10th-order* and *20th-order NARMA* datasets.

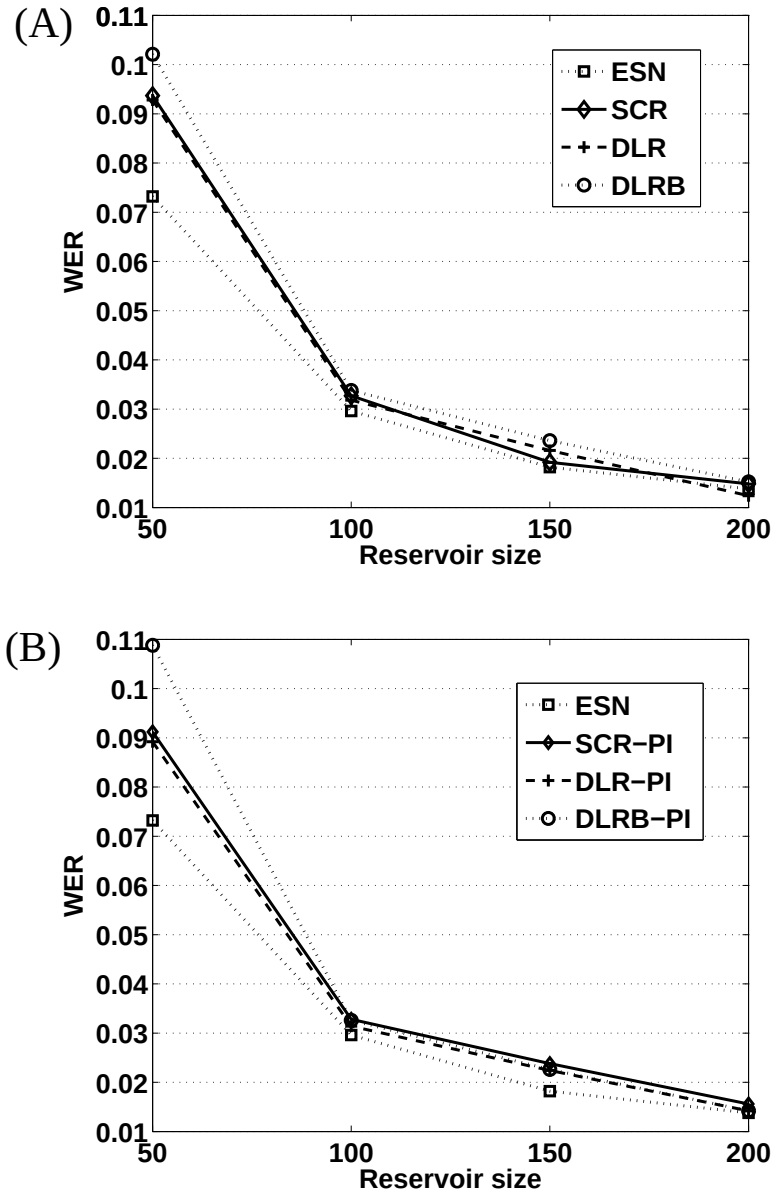

Figure 3.7: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *Isolated Digits* (speech recognition) task using two ways of generating input connection sign patterns; using random generation (i.i.d. Bernoulli distribution with mean 1/2) (A), and initial digits of π (B). Reservoir nodes with *tanh* transfer function f were used.

complex ESN reservoir structure is needed. Non-linearity in the reservoir activation and the reservoir size seem to be two important factors for successful learning on those three datasets. The differences in the results of ESN and SCR on *laser*, *Hénon Map* datasets were not statistically significant.

Figure 3.5 and 3.6 present results for the four model classes on the three *NARMA* time series, namely fixed *NARMA* of order 10, 20 and random *NARMA* of order 10. Figure 3.6 shows that the performance of linear reservoirs do not improve with increasing reservoir size. Interestingly, within the studied reservoir range (50-200), linear reservoirs beat the non-linear ones on *20-th order NARMA*. The situation changes for larger reservoir sizes. For example, non-linear ESN and SCR reservoirs of size 800 lead to the average NMSE of 0.0468 (std 0.0087) and 0.0926 (std 0.0039), respectively. For all *NARMA* series (see Figure 3.5), the SCR network is either the best performing architecture or is not worse than the best performing architecture in a statistically significant manner, where in case of *20-th order NARMA* with reservoir sizes of $N = 50$ and $N = 100$, SCR beats ESN at significance levels greater than 99%. It also beats ESN with reservoir size of $N = 150$ at significance level greater that 96%. Note that *NARMA* time series constitute one of the most important and widely used benchmark datasets used in the echo state network literature (e.g. (Schrauwen et al., 2008b; Cernansky and Tino, 2008; Jaeger, 2001, 2002a, 2003; Jaeger and Hass, 2004; Verstraeten et al., 2007; Steil, 2007)).

The results for the high-dimensional data set *Isolated Digits* are presented in figure 3.7(A). Except for the reservoir size 50, the performances of all studied reservoir models are comparable but not statistically significance (see table A.12 in [Appendix A]). When compared to ESN, the simplified reservoir models seem to work equally well on this high dimensional input series.

For *IPIX Radar*, *Sunspot Series* and *Non-linear System with Observational Noise* the results are presented in tables 3.1 and 3.2, respectively. On these data sets, the ESN performance did not always monotonically improve with the increasing reservoir size.

That is why for each data set we determined the best performing ESN reservoir size on the validation set ($N = 80$, $N = 200$, $N = 100$ for *IPIX Radar*, *Sunspot* Series and *Non-linear System with Observational Noise*, respectively). The performance of the other model classes (DLR, DLRB and SCR) with those reservoir sizes was then compared to that of ESN. In line with most RC studies using the *Sunspot* data set (e.g. (Schwenker and Labib, 2009)), we found that linear reservoirs were on a par and sometimes better (within the range of reservoir sizes considered in our experiments) with the non-linear ones. For all three data sets, the SCR architecture perform better than standard ESN, where the differences are in most cases highly statistical significant at levels greater than 99.8% (p values were smaller than 0.002). Except for *Non-linear System with Observational Noise* dataset when the variance $\sigma_v^2 = 1$, the differences in the results were not statistically significant.

Table 3.1: Mean NMSE for ESN, DLR, DLRB, and SCR across 10 simulation runs (standard deviations in parenthesis) on the *IPIX Radar* and *Sunspot* series. The results are reported for prediction horizon h and models with nonlinear reservoirs of size $N = 80$ (*IPIX Radar*) and linear reservoirs with $N = 200$ nodes (*Sunspot series*).

Data	h	ESN	DLR	DLRB	SCR
Radar	1	0.00115 (2.48E-05)	0.00112 (2.03E-05)	0.00110 (2.74E-05)	0.00109 (1.59E-05)
	5	0.0301 (8.11E-04)	0.0293 (3.50E-04)	0.0296 (5.63E-04)	0.0291 (3.20E-04)
Sunspot	1	0.1042 (8.33E-5)	0.1039 (9.19E-05)	0.1040 (7.68E-05)	0.1039 (5.91E-05)

Table 3.2: Mean NMSE for ESN, DLR, DLRB, and SCR across 10 simulation runs (standard deviations in parenthesis) on the *Nonlinear System with Observational Noise* data set. Reservoirs had $N = 100$ internal nodes with *tanh* transfer function f.

var w	var v	ESN	DLR	DLRB	SCR
1	1	0.4910 (0.0208)	0.4959 (0.0202)	0.4998 (0.0210)	0.4867 (0.0201)
10	1	0.7815 (0.00873)	0.7782 (0.00822)	0.7797 (0.00631)	0.7757 (0.00582)
1	10	0.7940 (0.0121)	0.7671 (0.00945)	0.7789 (0.00732)	0.7655 (0.00548)
10	10	0.9243 (0.00931)	0.9047 (0.00863)	0.9112 (0.00918)	0.9034 (0.00722)

Ganguli, Huh and Sompolinsky (Ganguli et al., 2008) quantified and theoretically analysed memory capacity of non-autonomous linear dynamical systems (corrupted by a

Gaussian state noise) using Fisher information between the state distributions at distant times. They found that the optimal Fisher memory is achieved for so called non-normal networks with DLR or DLRB topologies and derived the optimal input weight vector for those linear reservoir architectures. We tried setting the input weights to the theoretically derived values, but the performance did not improve over our simple strategy of randomly picked signs of input weights followed by model selection on the validation set. Of course, the optimal input weight considerations of (Ganguli et al., 2008) hold for linear reservoir models only. Furthermore, according to (Ganguli et al., 2008), the linear SCR belongs to the class of so called normal networks which are shown to be inferior to the non-normal ones. Interestingly enough, in our experiments, the performance of linear SCR was not worse than that of non-normal networks.

3.2.4 Further Simplifications of Input Weight Structure

The only random element of the SCR architecture is the distribution of the input weight signs. We found that any attempt to impose a regular pattern on the input weight signs (e.g. a periodic structure of the form $+-+-...$, or $+--+--...$ etc.) led to performance deterioration. Interestingly enough, it appears to be sufficient to relate the sign pattern to a single *deterministically* generated aperiodic sequence. Any simple pseudo-random generation of signs with a fixed seed is fine. Such sign patterns worked universally well across all benchmark data sets used in this study. For demonstration, we generated the universal input sign patterns in two ways:

1. the input signs are determined from decimal expansion $d_0.d_1d_2d_3...$ of irrational numbers (in our case π (**PI**) and e (**EX**)). The first N decimal digits $d_1, d_2, ..., d_N$ are thresholded at 4.5, e.g. if $0 \leq d_n \leq 4$ or $5 \leq d_n \leq 9$, then the n-th input connection sign (linking the input to the n-th reservoir unit) will be $-$ or $+$, respectively,

2. (**Log**) - the input signs are determined by the first N iterates in binary symbolic

dynamics of the logistic map $f(x) = 4x(1-x)$ in a chaotic regime (initial condition was 0.33, generating partition for symbolic dynamics with cut-value at 1/2).

The results shown in figures 3.8 (*NARMA*, *laser*, *Hénon Map* and *Non-linear Communication Channel* data sets), 3.7(B) (*Isolated Digits*), and tables 3.3 and 3.4 (*IPIX Radar*, *Sunspot*, and *Non-linear System with Observational Noise*), indicate that comparable performances of our SCR topology can be obtained without any stochasticity in the input weight generation by consistent use of a *deterministically* generated 'pseudo-random' aperiodic input signs across a variety of data sets. The results of ESN and *deterministic* SCR on the *20-th order NARMA* and *Non-linear Communication Channel* data sets are statistically significant with significance levels greater than 99%. Detailed results are presented in tables A.13 : A.16 [Appendix A].

Table 3.3: NMSE for ESN (mean across 10 simulation runs, standard deviations in parenthesis) and SCR topologies with deterministic input sign generation on the *IPIX Radar* and *Sunspot* series. The results are reported for nonlinear reservoirs of size $N = 80$ (*IPIX Radar*) and linear reservoirs with $N = 200$ nodes (*Sunspot* series).

Dataset	prediction horizon	ESN	SCR-PI	SCR-EX	SCR-Log
IPIX Radar	1	0.00115 (2.48E-05)	0.00109	0.00109	0.00108
	5	0.0301 (8.11E-04)	0.0299	0.0299	0.0297
Sunspot	1	0.1042 (8.33E-5)	0.1063	0.1065	0.1059

Table 3.4: NMSE for ESN (mean across 10 simulation runs, standard deviations in parenthesis) and SCR topologies with deterministic input sign generation on the *Nonlinear System with Observational Noise*. Nonlinear reservoirs had $N = 100$ nodes.

var w	var v	ESN	SCR-PI	SCR-EX	SCR-Log
1	1	0.4910 (0.0208)	0.5011	0.5094	0.5087
10	1	0.7815 (0.00873)	0.7910	0.7902	0.7940
1	10	0.7940 (0.0121)	0.7671	0.7612	0.7615
10	10	0.9243 (0.00931)	0.8986	0.8969	0.8965

We tried to use these simple deterministic input sign generation strategy for the other simplified reservoir models (DLR and DLRB). The results were consistent with our

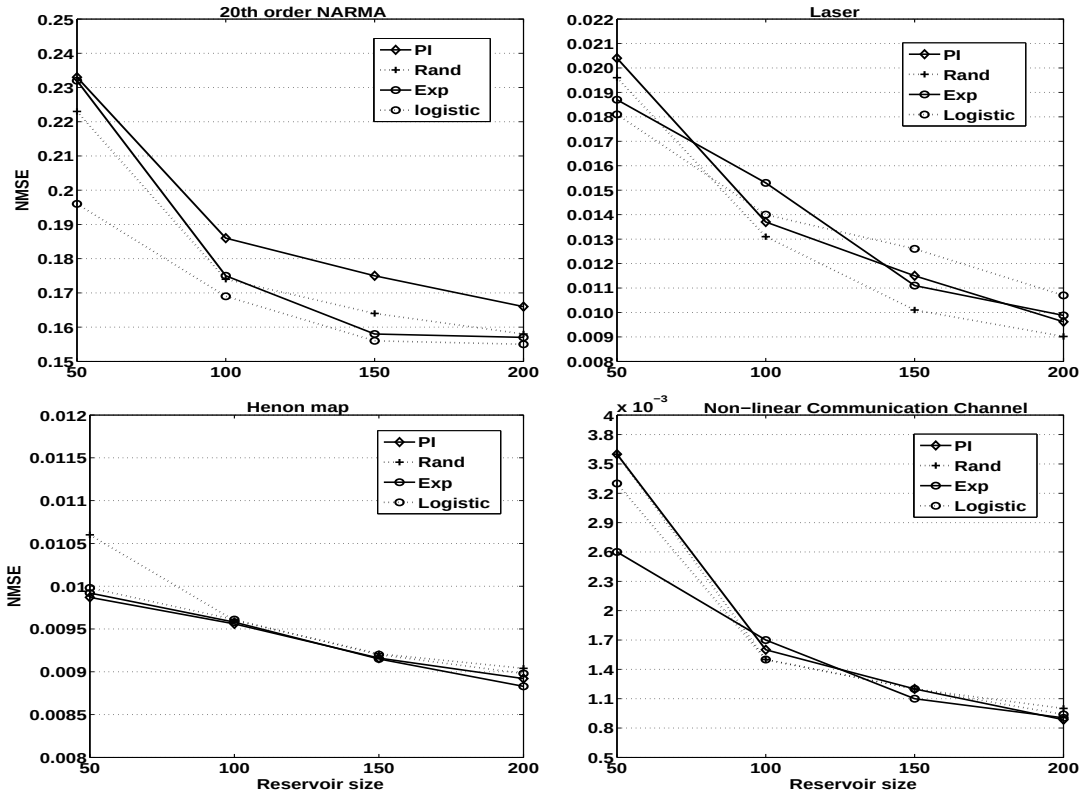

Figure 3.8: Test set performance of SCR topology using four different ways of generating pseudo-randomised sign patterns; using initial digits of π , and Exp ; logistic map trajectory, and random generation (i.i.d. Bernoulli distribution with mean 1/2). The result are reported for *20th NARMA*, *laser*, *Hénon Map*, and *Non-linear Communication Channel* datasets. Reservoir nodes with *tanh* transfer function f were used.

findings for the SCR. We also tried to simplify the input weight structure by connecting the input to a single reservoir unit only. However, this simplification either did not improve (e.g. NARMA dataset), or deteriorated the model performance (e.g. laser or Hénon Map).

3.2.5 Sensitivity Analysis

We tested the sensitivity of the model performance on 5-step ahead prediction with respect to variations in the (construction) parameters. The reservoir size is $N = 100$ for *NARMA* and *Laser* data sets; and $N = 80$ for the *IPIX Radar* data set.

In the case of ESN we varied the input scaling, as well as the spectral radius and connectivity of the reservoir matrix. In figures 3.9(A), 3.10(A) and 3.11(A) we show how the performance depends on the spectral radius and connectivity of the reservoir matrix. The input scaling is kept fixed at the optimal value determined on the validation set. Performance variation with respect to changes in input scaling (while connectivity and spectral radius are kept fixed at their optimal values) are reported in table 3.5.

For the SCR and DLR models figures 3.9(C,D), 3.10(C,D) and 3.11(C,D) illustrate the performance sensitivity with respect to changes in the only two free parameters - the input and reservoir weights v and r, respectively.

In the case of DLRB model, figures 3.9(B), 3.10(B) and 3.11(B) present the performance sensitivity with respect to changes in the reservoir weights r and b, while keeping the input weight fixed to the optimal value.

All the studied reservoir models show robustness with respect to small (construction) parameter fluctuations around the optimal parameter setting.

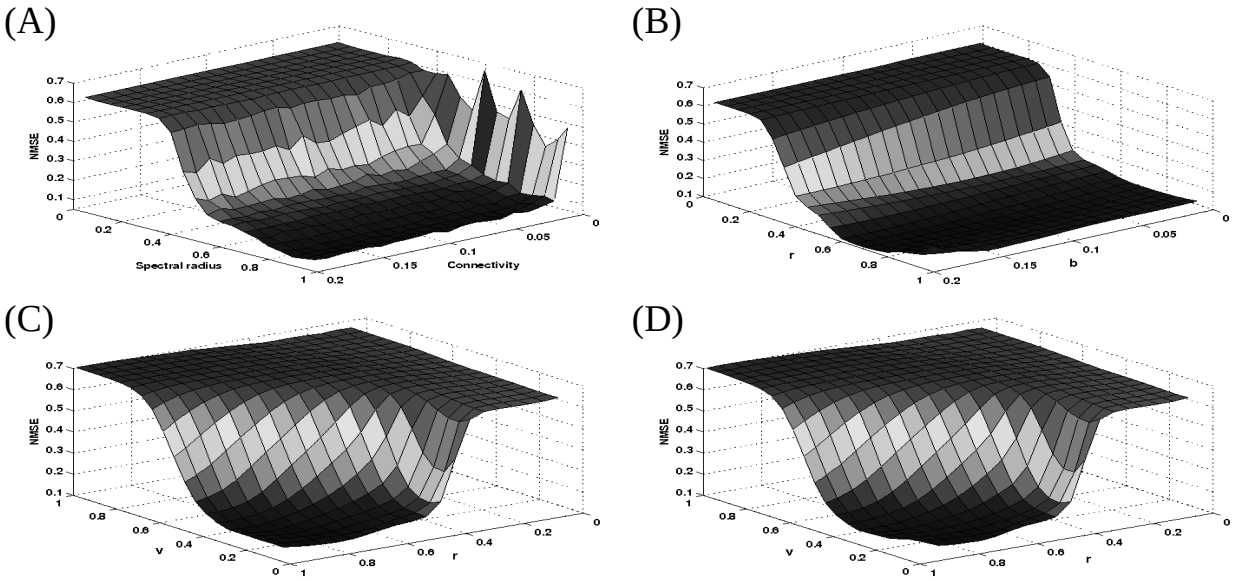

Figure 3.9: Sensitivity of ESN (A), DLRB (B), DLR (C), and SCR (D) topologies on the *10th order NARMA* dataset. The input sign patterns for SCR, DLR, and DLRB non-linear reservoirs were generated using initial digits of π.

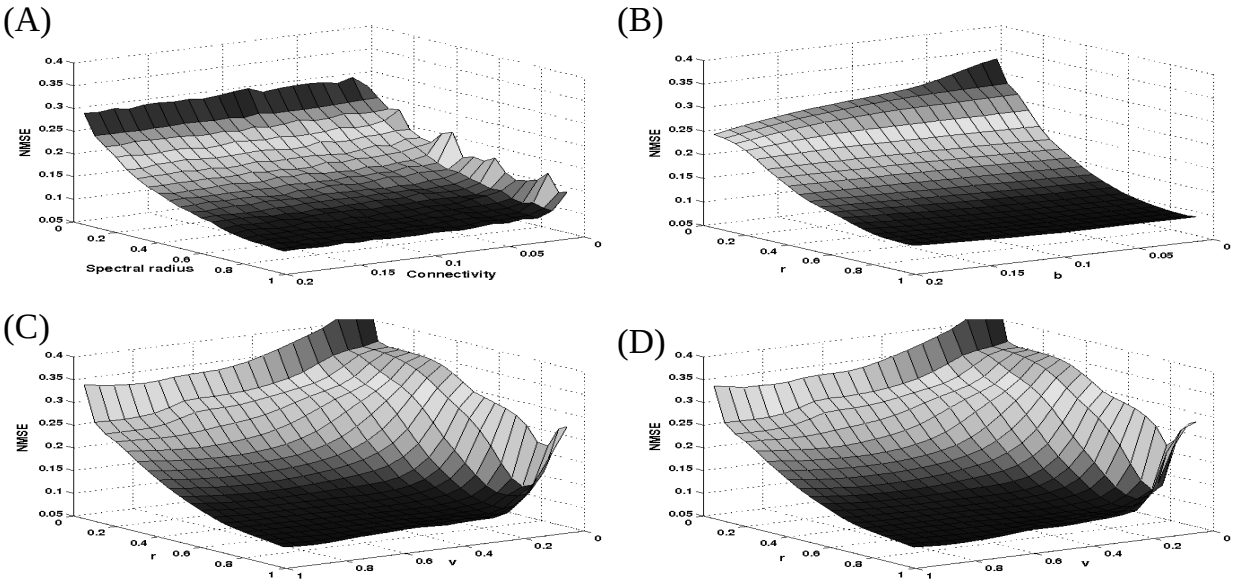

Figure 3.10: Sensitivity of ESN (A), DLRB (B), DLR (C), and SCR (D) topologies on the *laser* dataset. The input sign patterns for SCR, DLR, and DLRB non-linear reservoirs were generated using initial digits of π.

3.3 Short-term Memory Capacity of SCR Architecture

Jaeger (2002a) proved that for *any* recurrent neural network with N recurrent neurons,

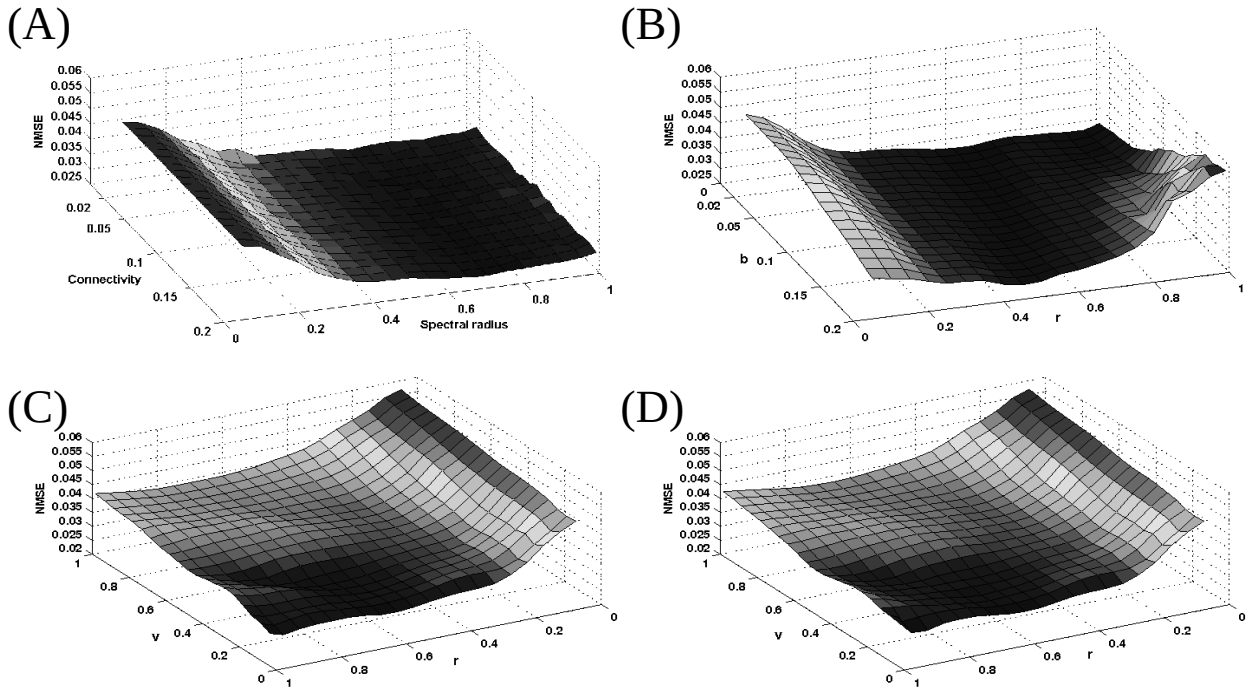

Figure 3.11: Sensitivity of ESN (A), DLRB (B), DLR (C), and SCR (D) topologies on the *IPIX Radar* dataset. The input sign patterns for SCR, DLR, and DLRB non-linear reservoirs were generated using initial digits of π.

under the assumption of i.i.d. input stream, the Short-term memory (STM) Capacity cannot exceed N. We prove (under the assumption of zero-mean i.i.d. input stream) that the STM capacity of linear SCR architecture with N reservoir units can be made arbitrarily close to N. Since there is a single input (univariate time series), the input matrix V is an N-dimensional vector $V = (V_1, V_2, ..., V_N)^T$.

Consider a vector rotation operator rot_1 that cyclically rotates vectors by 1 place to the right, e.g. $\text{rot}_1(V) = (V_N, V_1, V_2, ..., V_{N-1})^T$. For $k \geq 1$, the k-fold application of rot_1 is denoted by rot_k. The $N \times N$ matrix with k-th column equal to $\text{rot}_k(V)$ is denoted by Ω, e.g. $\Omega = (\text{rot}_1(V), \text{rot}_2(V), ..., \text{rot}_N(V))$.

Theorem 3.3.1 *Consider a linear SCR network with reservoir weight* $0 < r < 1$ *and an input weight vector* V *such that the matrix* Ω *is regular. Then the SCR network memory capacity is equal to*

$$MC = N - (1 - r^{2N}).$$

Table 3.5: Best connectivity and spectral radius for ESN with different input scaling for *10th order NARMA*, *laser* and *IPIX Radar* datasets.

Data set	Inp	Con	Spec	NMSE
10th order *NARMA*	0.05	0.18	0.85	0.1387 (0.0101)
	0.1	0.18	0.85	0.1075 (0.0093)
	0.5	0.18	0.85	0.2315 (0.0239)
	1	0.18	0.85	0.6072 (0.0459)
Laser	0.05	0.08	0.99	0.2738 (0.0128)
	0.1	0.08	0.99	0.1827 (0.0222)
	0.5	0.08	0.99	0.1058 (0.0070)
	1	0.08	0.99	0.0983 (0.0064)
IPIX Radar	0.05	0.2	0.7	0.0297 (0.00043)
	0.1	0.2	0.7	0.0311 (0.00087)
	0.5	0.2	0.7	0.0341 (0.0010)
	1	0.2	0.7	0.0378 (0.0014)

3.3.1 Notation and auxiliary results

We consider ESN with linear reservoir with cycle topology (SCR). The reservoir weight is denoted by r. Since we consider a single input, the input matrix V is an N-dimensional vector $V_{1..N} = (V_1, V_2, ..., V_N)^T$. By $V_{N..1}$ we denote the 'reverse' of $V_{1..N}$, e.g. $V_{N..1} = (V_N, V_{N-1}, ..., V_2, V_1)^T$.

Consider a vector rotation operator rot_1 that cyclically rotates vectors by 1 place to the right, e.g. given a vector $a = (a_1, a_2, ..., a_n)$, $\mathrm{rot}_1(a) = (a_n, a_1, a_2, ..., a_{n-1})$. For $k \geq 0$, the k-fold application of rot_1 is denoted by rot_k, where rot_0 is the identity mapping.

The $N \times N$ matrix with k-th column equal to $\mathrm{rot}_k(V_{N..1})$ is denoted by Ω, e.g.

$$\Omega = (\mathrm{rot}_1(V_{N..1}), \mathrm{rot}_2(V_{N..1}), ..., \mathrm{rot}_N(V_{N..1})).$$

We will need a diagonal matrix with diagonal elements $1, r, r^2, ..., r^{N-1}$:

$$\Gamma = \mathrm{diag}(1, r, r^2, ..., r^{N-1}).$$

Furthermore, we will denote the matrix $\Omega^T\ \Gamma^2\ \Omega$ by A,

$$A = \Omega^T\ \Gamma^2\ \Omega$$

and (provided A is invertible)

$$(\mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}))^T\ A^{-1}\ \mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}), \quad k \geq 0,$$

by ζ_k.

Lemma 3.3.2 *If Ω is a regular matrix, then $\zeta_N = 1$ and $\zeta_k = r^{-2k}$, $k = 1, 2, ..., N-1$.*

Proof: Denote the standard basis vector $(1, 0, 0, ..., 0)^T$ in $\Re^N$ by e_1. It holds:

$$\mathrm{rot}_k(V_{1..N}) = \Omega^T \mathrm{rot}_k(e_1), \quad k = 1, 2, ..., N-1.$$

This can be easily shown, as $\Omega^T \mathrm{rot}_k(e_1)$ selects the $(k+1)$st column of Ω^T ($(k+1)$st row of Ω), which is formed by $(k+1)$st elements of vectors $\mathrm{rot}_1(V_{N..1})$, $\mathrm{rot}_2(V_{N..1})$, ..., $\mathrm{rot}_N(V_{N..1})$. This vector is equal to the k-th rotation of $V_{1..N}$.

It follows that for $k = 1, 2, ..., N-1$,

$$(\mathrm{rot}_k(V_{1..N}))^T \Omega^{-1} = (\mathrm{rot}_k(e_1))^T$$

and so

$$\begin{aligned}
\zeta_k &= (\mathrm{rot}_k(V_{1..N}))^T\ A^{-1}\ \mathrm{rot}_k(V_{1..N}) \\
&= (\mathrm{rot}_k(V_{1..N}))^T\ \Omega^{-1}\ \Gamma^{-2}\ (\Omega^{-1})^T\ \mathrm{rot}_k(V_{1..N}) \\
&= (\mathrm{rot}_k(e_1))^T\ \Gamma^{-2}\ \mathrm{rot}_k(e_1). \\
&= r^{-2k}.
\end{aligned}$$

3.3.2 Proof of theorem 3.3.1

Given an i.i.d. zero-mean real-valued input stream $s(..t) = ... \; s(t-2) \; s(t-1) \; s(t)$ emitted by source P, the activations of the reservoir units at time t are given by

$$\begin{aligned}
x_1(t) &= V_1 \; s(t) + r \; V_N \; s(t-1) + r^2 \; V_{N-1} \; s(t-2) + r^3 \; V_{N-2} \; s(t-3) + ... \\
&... \; + r^{N-1} \; V_2 \; s(t-(N-1)) + r^N \; V_1 \; s(t-N) + r^{N+1} \; V_N \; s(t-(N+1)) + ... \\
&+ \; r^{2N-1} \; V_2 \; s(t-(2N-1)) + r^{2N} \; V_1 \; s(t-2N) + r^{2N+1} \; V_N \; s(t-(2N+1)) + ...
\end{aligned}$$

$$\begin{aligned}
x_2(t) &= V_2 \; s(t) + r \; V_1 \; s(t-1) + r^2 \; V_N \; s(t-2) + r^3 \; V_{N-1} \; s(t-3) + ... \\
&+ \; r^{N-1} \; V_3 \; s(t-(N-1)) + r^N \; V_2 \; s(t-N) + r^{N+1} \; V_1 \; s(t-(N+1)) + ... \\
&+ \; r^{2N-1} \; V_3 \; s(t-(2N-1)) + r^{2N} \; V_2 \; s(t-2N) + r^{2N+1} \; V_1 \; s(t-(2N+1)) \\
&+ \; r^{2N+2} \; V_N \; s(t-(2N+2)) + ...
\end{aligned}$$

...

$$\begin{aligned}
x_N(t) &= V_N \; s(t) + r \; V_{N-1} \; s(t-1) + r^2 \; V_{N-2} \; s(t-2) + ... + r^{N-1} \; V_1 \; s(t-(N-1)) \\
&+ \; r^N \; V_N \; s(t-N) + r^{N+1} \; V_{N-1} \; s(t-(N+1)) + ... + r^{2N-1} \; V_1 \; s(t-(2N-1)) \\
&+ \; r^{2N} \; V_N \; s(t-2N) + r^{2N+1} \; V_{N-1} \; s(t-(2N+1)) \\
&+ \; r^{2N+2} \; V_{N-2} \; s(t-(2N+2)) + ...
\end{aligned}$$

For the task of recalling the input from k time steps back, the optimal least-squares

readout vector U is given by

$$U = R^{-1} p_k, \tag{3.9}$$

where

$$R = E_{P(s(..t))}[x(t)x^T(t)]$$

is the covariance matrix of reservoir activations and

$$p_k = E_{P(s(..t))}[x(t)s(t-k)].$$

The covariance matrix R can be obtained in an analytical form. For example, because of the zero-mean and i.i.d. nature of the source P, the element $R_{1,2}$ can be evaluated as follows:

$$\begin{aligned}
R_{1,2} &= E_{P(s(..t))}[x(t)x^T(t)] \\
&= E[\ V_1\ V_2\ s^2(t) + r^2\ V_N\ V_1\ s^2(t-1) + r^4\ V_{N-1}\ V_N\ s^2(t-2) + ... \\
&+ r^{2(N-1)}\ V_2\ V_3\ s^2(t-(N-1)) + r^{2N}\ V_1\ V_2\ s^2(t-N) \\
&+ r^{2(N+1)}\ V_N\ V_1\ s^2(t-(N+1)) + ... + r^{2(2N-1)}\ V_2\ V_3\ s^2(t-(2N-1)) \\
&+ r^{4N}\ V_1\ V_2\ s^2(t-2N) + ...\ ] \\
&= V_1\ V_2\ Var[s(t)] + r^2\ V_N\ V_1\ Var[s(t-1)] + r^4\ V_{N-1}\ V_N\ Var[s(t-2)] + ... \\
&... + r^{2N}\ V_1\ V_2\ Var[s(t-N)] + ... \\
&= \sigma^2\ (V_1V_2 + r^2V_NV_1 + r^4V_{N-1}V_N + ... + r^{2(N-1)}V_2V_3 + r^{2N}V_1V_2 + ...) \\
&= \sigma^2\ (V_1V_2 + r^2V_NV_1 + r^4V_{N-1}V_N + ... + r^{2(N-1)}V_2V_3)\ \sum_{j=0}^{\infty} r^{2Nj}.
\end{aligned} \tag{3.10}$$

where σ^2 is the variance of P.

The expression (3.10) for $R_{1,2}$ can be written in a compact form as

$$R_{1,2} = \frac{\sigma^2}{1-r^{2N}}\ (\mathrm{rot}_1(V_{N..1}))^T\ \Gamma^2\ \mathrm{rot}_2(V_{N..1}). \tag{3.11}$$

In general,

$$R_{i,j} = \frac{\sigma^2}{1 - r^{2N}} \; (\mathrm{rot}_i(V_{N..1}))^T \; \Gamma^2 \; \mathrm{rot}_j(V_{N..1}), \quad i, j = 1, 2, ..., N, \tag{3.12}$$

and

$$R = \frac{\sigma^2}{1 - r^{2N}} \; \Omega^T \; \Gamma^2 \; \Omega. \tag{3.13}$$

By analogous arguments,

$$p_k = r^k \; \sigma^2 \; \mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}). \tag{3.14}$$

Hence, the optimal readout vector reads (see (3.9)):

$$U = (1 - r^{2N}) \; r^k \; A^{-1} \; \mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}). \tag{3.15}$$

The ESN output at time t is

$$\begin{aligned} y(t) &= x(t)^T \; U \\ &= (1 - r^{2N}) \; r^k \; x(t)^T \; A^{-1} \; \mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}). \end{aligned}$$

Covariance of the ESN output with the target can be evaluated as:

$$\begin{aligned} Cov(y(t), s(t-k)) &= (1 - r^{2N}) \; r^k \; Cov(x(t)^T, s(t-k)) \; A^{-1} \; \mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}) \\ &= r^{2k} \; (1 - r^{2N}) \; \sigma^2 \; (\mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}))^T \; A^{-1} \; \mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}) \\ &= r^{2k} \; (1 - r^{2N}) \; \sigma^2 \; \zeta_k. \end{aligned}$$

Variance of the ESN output is determined as:

$$\begin{aligned} Var(y(t)) &= U^T\ E[x(t)\ x(t)^T]\ U \\ &= U^T\ R\ U \\ &= p_k^T\ R^{-1}\ p_k \\ &= r^{2k}\ (\sigma^2)^2\ (\mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}))^T\ R^{-1}\ \mathrm{rot}_{k(\mathrm{mod})N}(V_{1..N}) \\ &= Cov(y(t), s(t-k)). \end{aligned} \tag{3.16}$$

We can now calculate the squared correlation coefficient between the desired output (input signal delayed by k time steps) and the network output $y(n)$:

$$\begin{aligned} MC_k &= \frac{Cov^2(s(t-k), y(t))}{Var(s(t))\ Var(y(t))} \\ &= \frac{Var(y(t)}{\sigma^2} \\ &= r^{2k}\ (1 - r^{2N})\ \zeta_k. \end{aligned}$$

The memory capacity of the ESN is given by

$$MC = MC_{\geq 0} - MC_0,$$

where

$$\begin{aligned} MC_{\geq 0} &= \sum_{k=0}^{\infty} MC_k \\ &= (1 - r^{2N})\ \left[\sum_{k=0}^{N-1} r^{2k}\ \zeta_k + \sum_{k=N}^{2N-1} r^{2k}\ \zeta_k + \sum_{k=2N}^{3N-1} r^{2k}\ \zeta_k + ...\ \right] \\ &= (1 - r^{2N})\ \left[\sum_{k=0}^{N-1} r^{2k}\ \zeta_k\right]\ \left[\sum_{k=0}^{\infty} r^{2k}\right] \\ &= \sum_{k=0}^{N-1} r^{2k}\ \zeta_k. \end{aligned}$$

Hence,

$$\begin{aligned} MC &= \sum_{k=0}^{N-1} r^{2k}\ \zeta_k - (1 - r^{2N})\zeta_0 \\ &= \zeta_0\ [1 - (1 - r^{2N})]\ + \sum_{k=1}^{N-1} r^{2k}\ \zeta_k \\ &= \zeta_0\ r^{2N}\ + \sum_{k=1}^{N-1} r^{2k}\ \zeta_k \\ &= \zeta_N\ r^{2N}\ + \sum_{k=1}^{N-1} r^{2k}\ \zeta_k \\ &= \sum_{k=1}^{N} r^{2k}\ \zeta_k. \end{aligned}$$

By lemma 3.3.2, $r^{2k}\ \zeta_k = 1$ for $k = 1, 2, ..., N-1$, and $r^{2N}\ \zeta_N = r^{2N}$. It follows that $MC = N - 1 + r^{2N}$.

3.3.3 Empirical Memory Capacity

We empirically evaluated the short-term memory capacity (MC) of ESN and our three simplified topologies. The networks were trained to memorise the inputs delayed by $k = 1, 2, ..., 40$. We used one input node, 20 linear reservoir nodes, and 40 output nodes (one for each k). The input consisted of random values sampled from a uniform distribution in the range [-0.5, 0.5]. The input weights for ESN and our simplified topologies have the same absolute value 0.5 with randomly selected signs. The elements of the recurrent weight matrix are set to 0 (80% of weights), 0.47 (10% of weights), or -0.47 (10% of weights), with 0.2 reservoir weights connection fraction and spectral radius $\lambda = 0.9$ (Ozturk et al., 2007). DLR and SCR weight r was fixed and set to the value $r = 0.5$. For DLRB $r = 0.5$ and $b = 0.05$. The output weights were computed using pseudo-inverse solution. The empirically determined MC values for ESN, DLR, DLRB and SCR models were (averaged over 10 simulation runs, standard dev. in parenthesis) 18.25 (1.46), 19.44 (0.89), 18.42

(0.96) and 19.48 (1.29), respectively. Note that the empirical MC values for linear SCR are in good agreement with the theoretical value of $20 - (1 - 0.5^{40}) \approx 19$.

3.3.4 Discussion

Jaeger (2002a) argues that if the vectors W^iV, $i = 1, 2, ..., N$, are linearly independent, then the memory capacity MC of linear reservoir with N units is N. Note that for the SCR reservoir

$$\mathrm{rot}_k(V) = \frac{W^kV}{r^k}, \quad k = 1, 2, ..., N,$$

and so the condition that W^iV, $i = 1, 2, ..., N$, are linearly independent directly translates into the requirement that the matrix Ω is regular. As $r \to 1$, the MC of SCR indeed approaches the optimal memory capacity N. According to Theorem 3.3.1, the MC measure depends on the spectral radius of W (in our case, r). Interestingly enough, in the verification experiments of (Jaeger, 2002a) with a reservoir of size $N = 20$ and reservoir matrix of spectral radius 0.98, the empirically obtained MC value was 19.2. Jaeger commented that a conclusive analysis of the disproportion between the theoretical and empirical values of MC was not possible, however, he suggested that the disproportion may be due to numerical errors, as the condition number of the reservoir weight matrix W was about 50. Using our result, $MC = N - (1 - r^{2N})$ with $N = 20$ and $r = 0.98$ yields $MC = 19.4$. It is certainly true that for smaller spectral radius values, the empirically estimated MC values of linear reservoirs decrease, as verified in several studies (e.g. (Verstraeten et al., 2007)), and this may indeed be at least partially due to numerical problems in calculating higher powers of W. Moreover, empirical estimates of MC tend to fluctuate rather strongly, depending on the actual i.i.d. driving stream used in the estimation (see e.g. (Ozturk et al., 2007)). Even though Theorem 3.3.1 suggests that the spectral radius of W should have an influence on the MC value for linear reservoirs, its influence becomes negligible for large reservoirs, since (provided Ω is regular) the MC of SCR is provably bounded within the interval $(N - 1, N)$.

Memory capacity MC of a reservoir is a representative member from the class of reservoir measures that quantify the amount of information that can be preserved in the reservoir about the past. For example, Ganguli, Huh and Sompolinsky (Ganguli et al., 2008) proposed a different (but related) quantification of memory capacity for linear reservoirs (corrupted by a Gaussian state noise). They evaluated the Fisher information between the reservoir activation distributions at distant times. Their analysis shows that the optimal Fisher memory is achieved for the reservoir topologies corresponding e.g. to our DLR or DLRB reservoir organisations. Based on the Fisher memory theory, the optimal input weight vector for those linear reservoir architectures was derived. Interestingly enough, when we tried setting the input weights to the theoretically derived values, the performance in our experiments did not improve over our simple strategy for obtaining the input weights. While in the setting of (Ganguli et al., 2008), the memory measure does not depend on the distribution of the source generating the input stream, the MC measure of (Jaeger, 2002a) is heavily dependent on the generating source. For the case of i.i.d. source (where no dependencies between the time series elements can be exploited by the reservoir) the memory capacity $MC = N - 1$ can be achieved by a very simple model: DLR reservoir with unit weight $r = 1$, one input connection with weight 1 connecting the input with the 1st reservoir unit, and for $k = 1, 2, ..., N - 1$ one output connection of weight 1 connecting the $(k+1)$-th reservoir unit with the output. The linear SCR, on the other hand, can get arbitrarily close to the theoretical limit $MC = N$. In cases of non i.i.d. sources, the temporal dependencies in the input stream can increase the memory capacity beyond the reservoir size N (Jaeger, 2002a).

3.4 Chapter Summary

Throughout this chapter, Simple Echo Sate Network (ESN) Architectural Designs have been introduced. It has also been presented that for a variety of tasks it is sufficient to consider:

1. a simple fixed non-random cycle reservoir topology with full connectivity from inputs to the reservoir (SCR) ,

2. a single fixed absolute weight value r for all reservoir connections and

3. a single weight value v for input connections, with *deterministically* generated "pseudo-random" aperiodic pattern of input signs.

A simple *deterministically* constructed cycle reservoir (SCR) is comparable to the standard echo state network methodology. The (short term) memory capacity of linear cyclic reservoirs can be made arbitrarily close to the proved optimal value.

Chapter 4

Cycle Reservoir with Regular Jumps

In chapter 3 we argued that randomisation and trail-and-error construction of reservoirs may not be necessary. Very simple, cyclic, deterministically generated reservoirs were shown to yield performance competitive with standard ESN on a variety of data sets of different origin and memory structure. We also proved that the memory capacity of linear Simple Cycle reservoir (SCR) can be made arbitrarily close to the proven optimal value (for any recurrent neural network of the ESN form). In this chapter we propose to extend SCR model in three aspects:

1. We introduce a novel simple deterministic reservoir model, Cycle Reservoir with Jumps (CRJ), with highly constrained weight values, that has superior performance to standard ESN on a variety of temporal tasks of different origin and characteristics.

2. We elaborate on the possible link between reservoir characterisations, such as eigenvalue distribution of the reservoir matrix or pseudo-Lyapunov exponent of the input-driven reservoir dynamics, and the model performance. It has been suggested that a uniform coverage of the unit disk by such eigenvalues can lead to superior model performances. We show that despite highly constrained eigenvalue distribution, CRJ consistently outperform ESN (that have much more uniform eigenvalue coverage of

the unit disk). Also, unlike in the case of ESN, pseudo-Lyapunov exponents of the selected 'optimal' CRJ models are consistently negative.

3. We present a new framework for determining short term memory capacity of linear reservoir models to a high degree of precision. Using the framework we study the effect of shortcut connections in the CRJ reservoir topology on its memory capacity.

In this chapter we extend the Simple Cycle Reservoir (SCR) introduced in chapter 3, with a regular structure of shortcuts (Jumps) - *Cycle Reservoir with Jumps* (CRJ). In the spirit of SCR we keep the reservoir construction simple and deterministic. Yet, it will be shown that such an extremely simple regular architecture can significantly outperform both SCR and standard randomised ESN models. Prompted by these results, we investigate some well known reservoir characterisations, such as eigenvalue distribution of the reservoir matrix, pseudo-Lyapunov exponent of the input-driven reservoir dynamics, or memory capacity and their relation to the ESN performance.

The chapter is organised as follows. Section 4.1 presents our proposed model - CRJ. Experimental results are presented and discussed in Sections 4.2 and 4.3, respectively. Section 4.4 investigates three reservoir characterisations (eigen-spectrum of the reservoir weight matrix, short term memory capacity and pseudo-Lyapunov exponent) in the context of reservoir models studied in this work. Finally, this chapter is summarised in section 4.5.

4.1 Cycle Reservoir with Jumps

In chapter 3 we proposed a Simple Cycle Reservoir (SCR) with performance competitive to that of standard ESN. Unlike ESN, the construction of SCR model is completely deterministic and extremely simple. All cyclic reservoir weights have the same value; all input connections also have the same absolute value. Viewing reservoir interconnection

topology as a graph, the SCR has a small degree of local clustering and a large average path length. In contrast, ESN (a kind of random network) has small degree of local clustering and small average path length. It has been argued that reservoirs should ideally have small clustering degree (sparse reservoirs) (Jaeger and Hass, 2004) so that the dynamic information flow through the reservoir nodes is not 'too cluttered'. Also a small average path length, while having longer individual paths within the reservoir, can allow for representation of a variety of dynamical time scales. We propose a Cycle Reservoir with Jumps (CRJ) which, compared with SCR leads to slightly higher degree of local clustering while achieving much smaller average path length.

The CRJ model has a fixed simple regular topology: the reservoir nodes are connected in a uni-directional cycle (as in SCR) with bi-directional shortcuts (jumps) (Fig. 4.1). All cycle connections have the same weight $r_c > 0$ and likewise all jumps share the same weight $r_j > 0$. In other words, non-zero elements of W are:

- the 'lower' sub-diagonal $W_{i+1,i} = r_c$, for $i = 1...N-1$,
- the 'upper-right corner' $W_{1,N} = r_c$ and
- the jump entries r_j. Consider the jump size $1 < \ell < \lfloor N/2 \rfloor$. If $(N \mod \ell) = 0$, then there are N/ℓ jumps, the first jump being from unit 1 to unit $1+\ell$, the last one from unit $N+1-\ell$ to unit 1 (see Figure 2 (A)). If $(N \mod \ell) \neq 0$, then there are $\lfloor N/\ell \rfloor$ jumps, the last jump ending in unit $N+1-(N \mod \ell)$ (see Figure 2 (B)). In such cases, we also consider extending the reservoir size by κ units ($1 \leq \kappa < \ell$), such that $(N+\kappa) \mod \ell = 0$. The jumps are bi-directional sharing the same connection weight r_j.

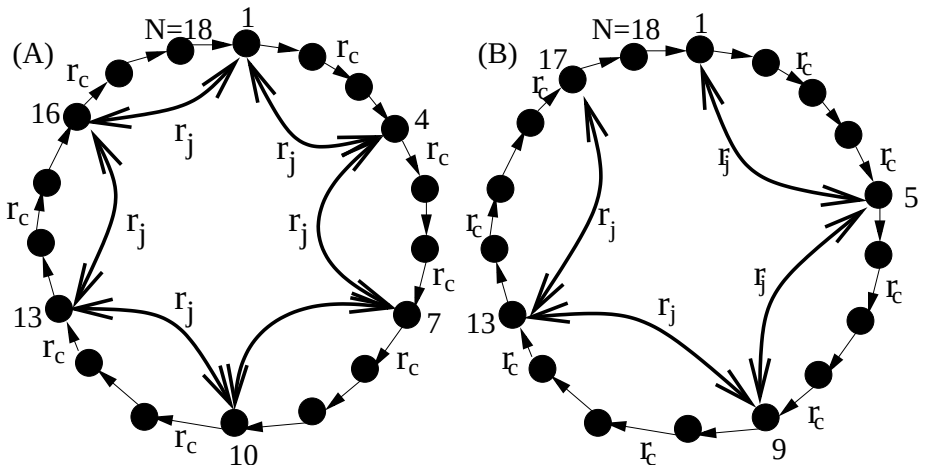

Figure 4.1: An example of CRJ reservoir architecture with $N = 18$ units and jump size $\ell = 3$ (A) and $\ell = 4$ (B).

As with the SCR model, in the CRJ model we use full input-to-reservoir connectivity with the same absolute value $v > 0$ of the connection weight. We showed in section 3.2.4 that an aperiodic character of signs of the input weights in $V = (V_1, V_2, ..., V_K)$ is essential for the SCR model. Conversely, in this chapter we use the same method for obtaining the input weight signs, universally across all data sets. In particular, the input signs are determined from decimal expansion $d_0.d_1d_2d_3...$ of an irrational number - in our case π. The first N decimal digits $d_1, d_2, ..., d_N$ are thresholded at 4.5, i.e. if $0 \leq d_n \leq 4$ and $5 \leq d_n \leq 9$, then the n-th input connection sign (linking the input to the n-th reservoir unit) will be $-$ and $+$, respectively. The values v, r_c, and r_j are chosen on the validation set.

4.2 Experiments

In this section we test and compare our simple CRJ reservoir topology with standard ESN and SCR on a variety of timeseries tasks widely used in the ESN literature and covering a wide spectrum of memory structure (Schrauwen et al., 2008b; Cernansky and Tino, 2008;

Jaeger, 2001, 2002a, 2003; Jaeger and Hass, 2004; Verstraeten et al., 2007; Steil, 2007).

4.2.1 Experimental Setup

For each data set and each model class (ESN, SCR, and CRJ) we picked on the validation set a model representative to be evaluated on the test set. The readout mapping was fitted both using offline (Ridge Regression) and online (RLS) training. Then, based on validation set performance, the offline or online trained readout was selected and tested on the test set. We present the results for three reservoir sizes $N = 100, 200, 300$.

- For RLS training we add noise to the internal reservoir activations where the noise is optimised for each dataset and each reservoir size using a validation set (Wyffels et al., 2008).
- For SCR architecture the model representative is defined by the absolute input weight value $v \in (0, 1]$ and the reservoir cycle connection weight $r_c \in (0, 1]$.
- For the CRJ architecture the model representative is defined by the absolute input weight value $v \in (0, 1]$, the reservoir cycle connection weight $r_c \in (0, 1]$, the jump size $1 < \ell < \lfloor N/2 \rfloor$ and the jump weight $r_j \in (0, 1]$.
- For the ESN architecture, the model representative is specified by the reservoir sparsity, spectral radius λ of the reservoir weight matrix, input weight connectivity and input weight range $[-a, a]$.

For ESN we calculated out-of sample (test set) performance measures over 10 simulation runs (presented as mean and StDev). The selected SCR and CRJ representatives are evaluated out-of-sample only once, since their construction is completely deterministic. The only exception is the speech recognition experiment - due to limited test set size, following (Verstraeten et al., 2007), a 10-fold cross-validation was performed (and paired t-test was used to assess statistical significance of the result).

Details of the experimental setup, including ranges for cross-validation based grid search on free-parameters, are presented in Table 4.1. Detailed parameter settings of the selected model representatives can be found in Appendix B.

Table 4.1: Summary of the experimental setup. Grid search ranges are specified in MATLAB notation, i.e. $[s : d : e]$ denotes a series of numbers starting from s, increased by increments of d, until the ceiling e is reached.

Reservoir topologies	ESN, SCR and CRJ
Readout learning	RLS with dynamic noise injection, Ridge Regression
Reservoir matrix	ESN (random weights with spectral radius α in $[0.05 : 0.05 : 1]$, and connectivity *con* in $[0.05 : 0.05 : 0.5]$) CRJ and SCR (r_c in $[0.05 : 0.05 : 1]$, r_j in $[0.05 : 0.05 : 1]$)
jump size	$1 < \ell < \lfloor N/2 \rfloor$, where N is the reservoir size.
reservoir size	N in $[100 : 100 : 300]$
input scale	v (for SCR and CRJ) and a (for ESN) from $[0.01 : 0.005 : 1]$
input sign generation	SCR and CRJ: thresholded decimal expansion of π
readout regularisation	reservoir noise size (RLS), regularisation factor (ridge regression) 10^q, $q = [-15 : 0.25 : 0]$

4.2.2 Experimental tasks

4.2.2.1 System Identification

As a System Identification task, we considered a 10th order NARMA system (Atiya and Parlos, 2000) which we described in detail in section 3.2.1.

NARMA sequence has a length of 8000 items where the first 2000 were used for training, the following 5000 for validation, and the remaining 2000 for testing. The first 200 values from the training, validation and test sequences were used as the initial washout period.

The results are presented in Table 4.2. Even though SCR is slightly inferior to the standard ESN construction, the simple addition of regular shortcuts (jumps) to the SCR leads to a superior performance of CRJ topology, where for reservoir size $N = 100$ the CRJ model is significantly superior to ESN at ($p \approx 0.000042$). For reservoirs with $N = 200$ and $N = 300$ neurons CRJ beats ESN at significance level 99.9%. Note that the significance levels where determined for CRJ by performing a different NARMA dataset at each run.

Table 4.2: Test Set NMSE Results of ESN, SCR, and CRJ Reservoir Models on the 10th Order NARMA System. Reservoir Nodes with *tanh* Transfer Function were Used.

reservoir model	$N = 100$	$N = 200$	$N = 300$
ESN	0.0788 (0.00937)	0.0531 (0.00198)	0.0246 (0.00142)
SCR	0.0868	0.0621	0.0383
CRJ	**0.0619**	**0.0196**	**0.0130**

4.2.2.2 Time Series Prediction

The Santa Fe Laser dataset (Jaeger et al., 2007a) is a cross-cut through periodic to chaotic intensity pulsations of a real laser. The task was to predict the next value $y(t + 1)$. The dataset contains 9000 values, the first 2000 values were used for training, the next 5000 for validation, and the remaining 2000 values was used for testing the models. The first 200 values from training, validation and testing sequences were used as the initial washout period.

The results are shown in Table 4.3. Again, ESN and SCR are almost on-par, with

SCR slightly inferior. However, the CRJ topology can outperform the other architectures by a large margin.

Table 4.3: Test Set NMSE Results of ESN, SCR, and CRJ Reservoir Models on the Santa Fe Laser Dataset. Reservoir Nodes with *tanh* Transfer Function were Used.

reservoir model	$N = 100$	$N = 200$	$N = 300$
ESN	0.0128 (0.00371)	0.0108 (0.00149)	0.00895 (0.00169)
SCR	0.0139	0.0112	0.0106
CRJ	**0.00921**	**0.00673**	**0.00662**

Figures 4.2 and 4.3 present one step-ahead prediction for laser time series using CRJ with reservoir size 200. Figure 4.2(A) shows the prediction curve, where it can be shown that it is difficult to predict the values when there is a cross-cut in the dataset (t in [70,80]). Figure 4.2(B) shows prediction error which is the difference between the predicted and target outputs. Moreover, figure 4.3(A) shows Predicted output and figure 4.3(B) presents traces of some selected units.

4.2.2.3 Speech Recognition

For this task we used the Isolated Digits dataset which is described in detail in chapter 3 Section 3.2.1. The dataset contains 500 spoken digits; because of the limited test set size, 10-fold cross-validation was performed (Verstraeten et al., 2007) and paired t-test was used to assess whether the perceived differences in model performance are statistically significant. Following the ESN literature using this dataset, the model performance will be evaluated using the Word Error Rate (WER). We use this data set to demonstrate the modelling capabilities of different reservoir models on high-dimensional (86 input channels) time series.

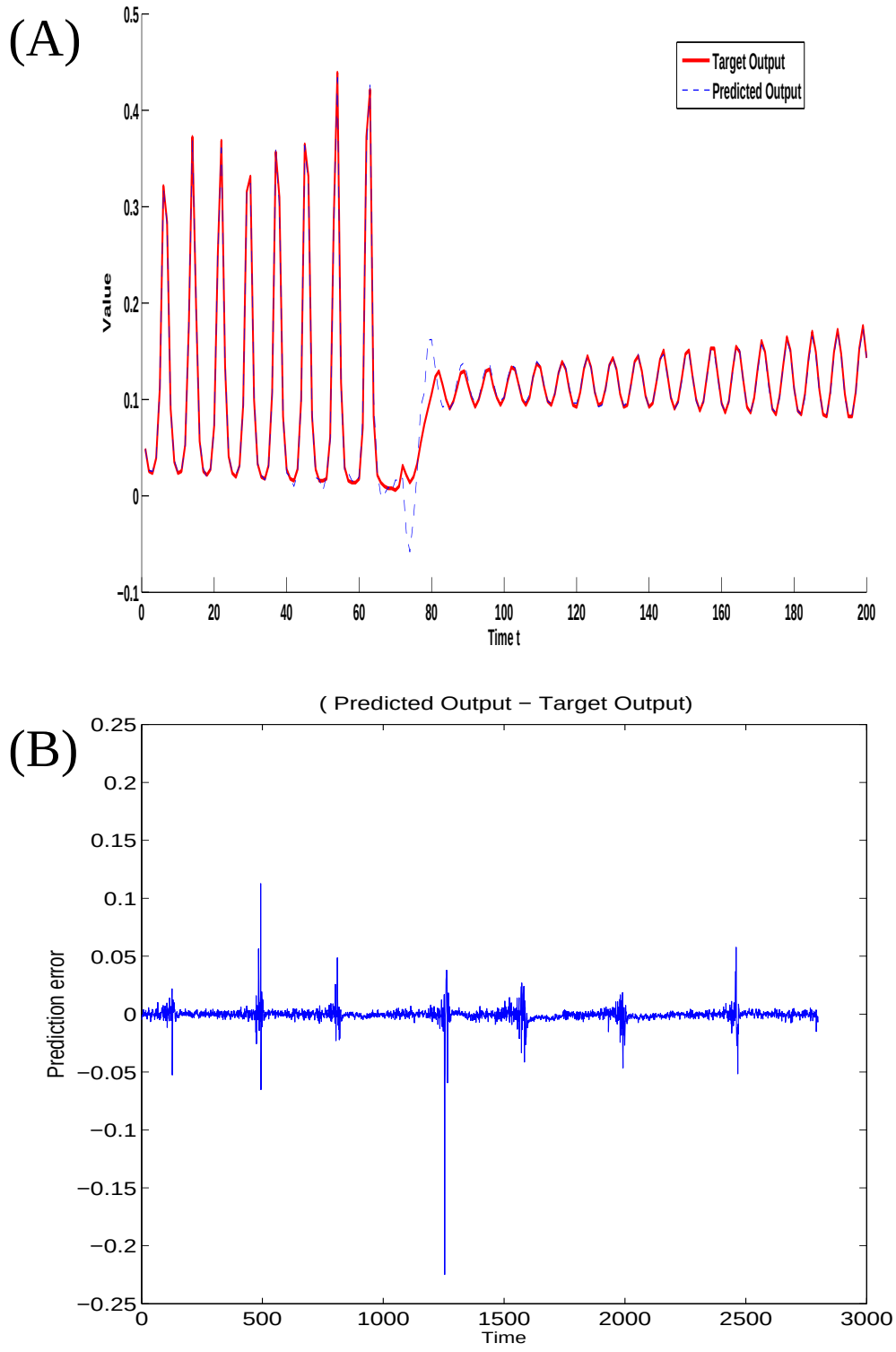

Figure 4.2: Single step-ahead prediction for laser time series using CRJ with reservoir size 200, prediction curve (A), and prediction error(B).

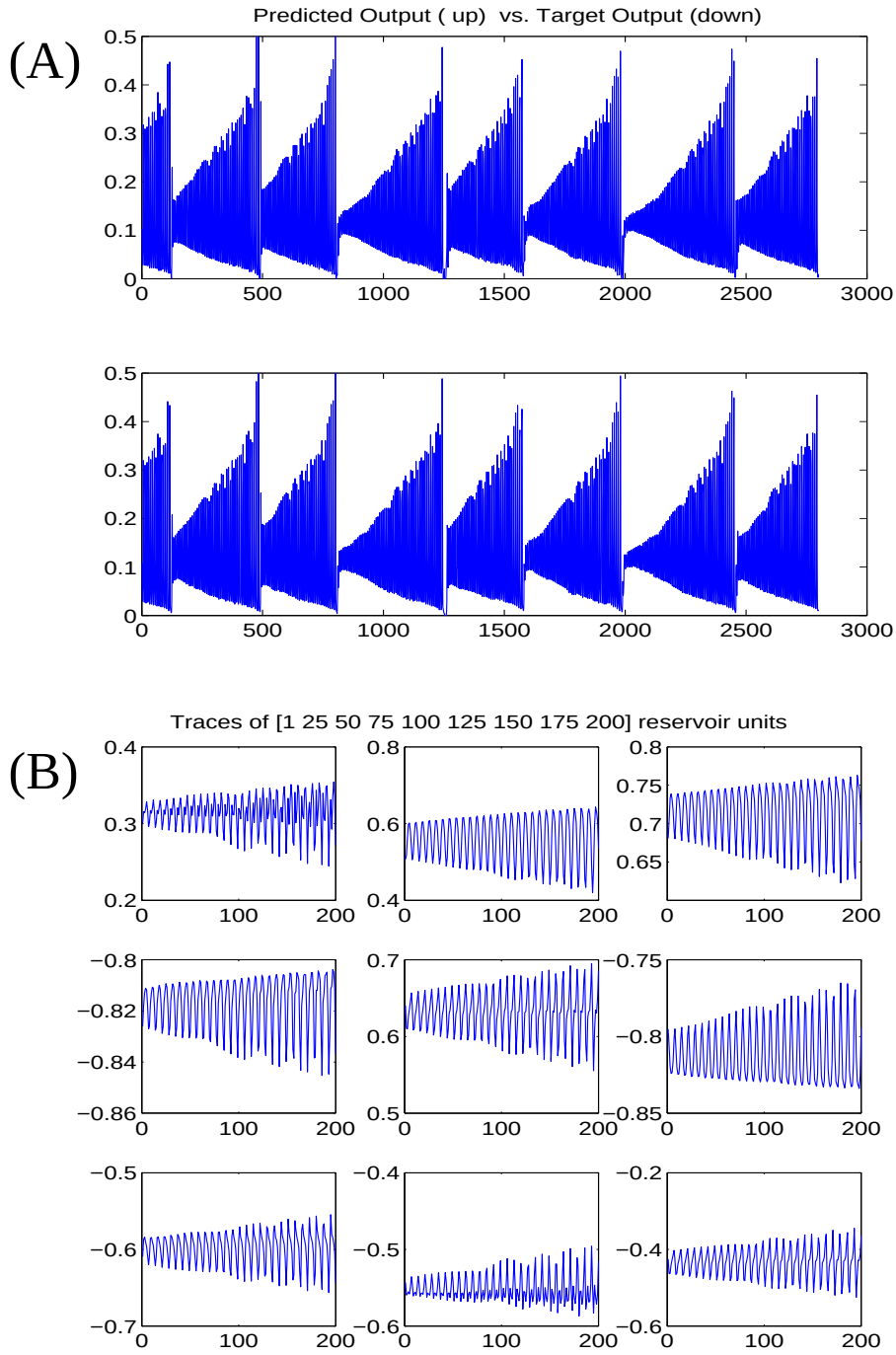

Figure 4.3: Predicted output time series vs. Target output time series (A), and Traces of some selected units of a 200-unit CRJ driven by the laser dataset (B) using CRJ with reservoir size 200.

The results confirming superior performance of the simple CRJ model are shown in Table 4.4. For reservoir size $N = 100$ the CRJ model is significantly superior to ESN at the confidence level 96%. For reservoirs with $N = 200$ and $N = 300$ neurons CRJ beats ESN at significance levels greater than 99%.

Table 4.4: WER Results of ESN, SCR, and CRJ Models on the *Isolated Digits* (Speech Recognition) Task. Reservoir Nodes with *tanh* Transfer Function f were Used.

reservoir model	$N = 100$	$N = 200$	$N = 300$
ESN	0.0296 (0.0063)	0.0138 (0.0042)	0.0092 (0.0037)
SCR	0.0329 (0.0031)	0.0156 (0.0035)	0.0081 (0.0022)
CRJ	**0.0281 (0.0032)**	**0.0117 (0.0029)**	**0.0046 (0.0021)**

4.2.2.4 Memory and Non-linear mapping task

The last task, used in (Verstraeten et al., 2010), is a generalisation of the delay XOR-task used in (Schrauwen et al., 2008a). It allows one to systematically study two characteristics of reservoir topologies: memory and the capacity to process non-linearities in the input time series. The memory is controlled by the delay d of the output, and the 'degree of non-linearity' is determined by a parameter $p > 0$. The input signal $s(t)$ contains uncorrelated values from a uniform distribution over the interval $[-0.8, 0.8]$. The task is to reconstruct a delayed and non-linear version of the input signal:

$$y_{p,d}(t) = sign[\beta(t-d)] \cdot |\beta(t-d)|^p, \tag{4.1}$$

where $\beta(t-d)$ is the product of two delayed successive inputs,

$$\beta(t-d) = s(t-d) \cdot s(t-d-1).$$

The sign and absolute values are introduced to assure a rotationally symmetric output even in the case of even powers (Verstraeten et al., 2010). Following (Verstraeten et al., 2010), we considered delays $d = 1, ..., 15$ and powers $p = 1, ..., 10$ with a total of 150 output signals $y_{p,d}$ (realised as 150 readout nodes). The main purpose of this experiment is to test whether a single reservoir can have rich enough pool of internal representations of the driving input stream so as to cater for the wide variety of of outputs derived from the input for a range of delay and non-linearity parameters.

We used time series of length 8000, where a new time series was generated in each of 10 runs. The first 2000 items were used for training, the next 3000 for validation, and the remaining 3000 for testing the models. The first 200 values from training, validation and test sequences were used as the initial washout period. As in (Verstraeten et al., 2010), we used reservoirs of size 100 nodes.

Figure 4.4 illustrates the NMSE performance for ESN (A) , SCR (B) and CRJ (C). Shown are contour plots across the two degrees of freedom – the delay d and the non-linearity parameter p. We also show difference plots between the respective NMSE values: ESN - SCR(D), ESN - CRJ (E) and SCR - CRJ (F). When the task becomes harder (non-linearity and delay increase - upper-right corner of the contour plots) the performance of the simple reservoir constructions, SCR and CRJ, is superior to that of standard ESN. Interestingly, the simple reservoirs seem to outperform ESN by the largest margin for moderate delays and weak non-linearity (small values of p). We do not have a clear explanation to offer but note that our later studies in section 4.4.2 show that, compared with ESN, the SCR and CRJ topologies have a potential for greater memory capacity. This seems to be reflected most strongly if the series is characterised by weak non-linearity.

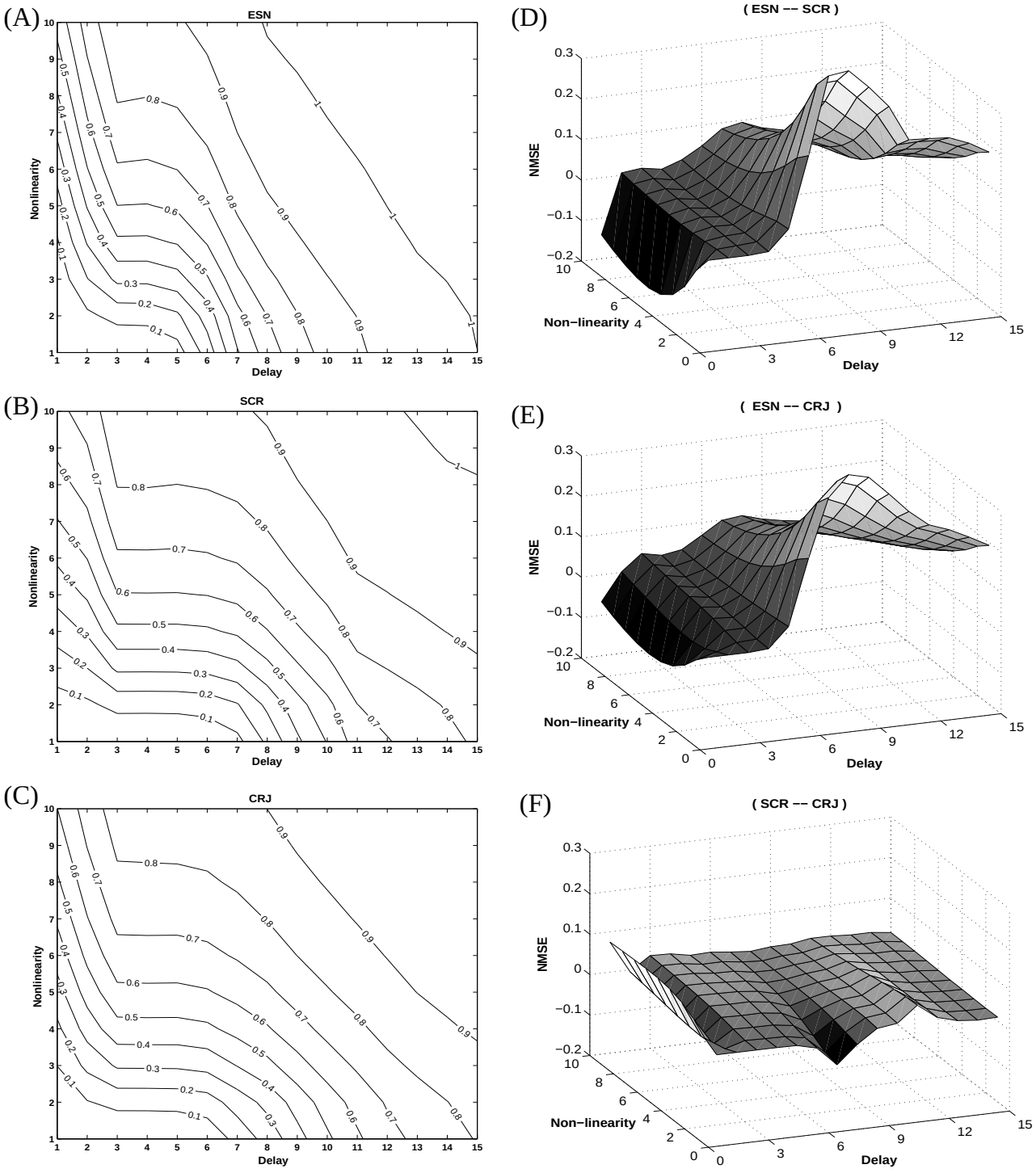

Figure 4.4: Memory and Non-Linear Mapping Task. Shown are NMSE Values for ESN (A), SCR (B) and CRJ (C). We also Show Difference Plots Between the Respective NMSE Values: ESN - SCR (D), ESN - CRJ (E) and SCR - CRJ (F).

4.3 Discussion

The experimental results clearly demonstrate that our very simple deterministic reservoir constructions have a potential to significantly outperform standard ESN randomised reservoirs. We propose that instead of relying on unnecessary stochastic elements in reservoir construction, one can obtain superior (and sometimes superior by a large margin) performance by employing the simple regular unidirectional circular topology with bi-directional jumps with fixed cycle and jump weights. However, it is still not clear exactly what aspects of dynamic representations in the reservoirs are of importance and why. In later sections we concentrate on three features of reservoirs - eigenspectrum of the reservoir weight matrix, (pseudo) Lyapunov exponent of the input-driven reservoir dynamics and short term memory capacity - and discuss their relation (or lack of) to the reservoir performance on temporal tasks.

Moreover, besides the symmetric bi-directional regular jumps (CRJ), we considered uni-directional jumps (both in the direction and in the opposite direction to the main reservoir cycle), as well as jumps not originating/ending in a regular grid of 'hub-like' nodes. For example, when a jump lands in unit n, the next jump originates in unit $n+1$ etc. In all cases, compared with our regular CRJ topology, the performance was slightly worse. However, when allowing for two different weight values in the bidirectional jumps (one for forward, one for backward jumps) (CRJ_{fb}), the performance improved slightly but not significantlyover CRJ (see table 4.5).

Table 4.5: NMSE for CRJ topologies using bi-directional jumps-CRJ, feedforward jumps-CRJ_f, backward jumps-CRJ_b, or feedforward & backward jumps-CRJ_{fb} on the laser time series using reservoir sizes of $N = 200, 500$.

CRJ model	$N = 200$	$N = 500$
CRJ	0.00673	0.00526
CRJ_{fb}	0.00638	0.00509
CRJ_f	0.00681	0.00512
CRJ_b	0.00645	0.00523

Our framework can be extended to more complex regular hierarchical reservoir constructions. For example, we can start with a regular structure of relatively short 'lower level' jumps in the style of CRJ topology. Then another layer of longer jumps over the shorter ones can be introduced etc. We refer to this architecture as *Cycle Reservoir with Hierarchical Jumps* (CRHJ). Figure 4.5 illustrates this idea on a 3-level hierarchy of jumps. As before, the cycle weights are denoted by r_c. The lowest level jump weights are denoted by r_{j_1}, the highest by r_{j_3}. On each hierarchy level, the jump weight has a single fixed value.

Table 4.6: Test Set NMSE Results of Deterministic CRHJ Reservoir Model on the Santa Fe Laser Dataset and NARMA System. Reservoir Nodes with *tanh* Transfer Function were Used.

Dataset	$N = 100$	$N = 200$	$N = 300$
laser	0.00743	0.00594	0.00581
NARMA	0.0662	0.0182	0.0133

As an illustrative example, in Table 4.6 we show test set results for 3-level jump hierarchies with jump sizes 4, 8 and 16. We used the same jump sizes for both laser and

NARMA data sets. The weights $r_c, r_{j_1}, r_{j_2}, r_{j_3} \in [0.05, 1)$ were found on the validation set. In most cases the performance of reservoirs with hierarchical jump structure slightly improves over the CRJ topology (see Tables 4.2 and 4.3). Detailed parameter settings of the selected model representatives can be found in Appendix B table B.3. However, such more complex reservoir constructions, albeit deterministic, diverge from the spirit of the simple SCR and CRJ constructions. The potential number of free parameters (jump sizes, jump weights) grows and the simple validation set search strategy can quickly become infeasible.

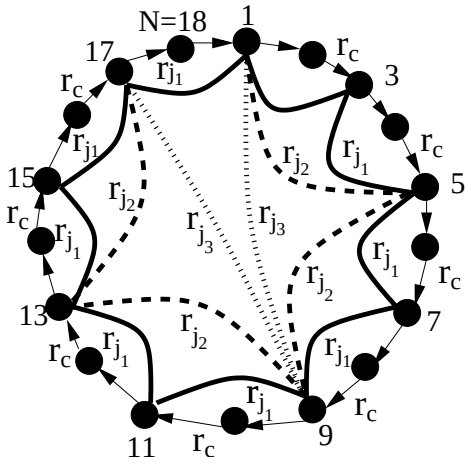

Figure 4.5: Reservoir Architecture of Cycle Reservoir with Hierarchical Jumps (CRHJ) with Three Hierarchical Levels. Reservoir Size $N = 18$, and the Jump Sizes are $\ell = 2$ for Level 1 , $\ell = 4$ for Level 2, and $\ell = 8$ for Level 3.

The CRHJ structure differs from hierarchically structured randomised reservoir models proposed in the RC community (Jaeger, 2007; Triefenbach et al., 2010), where the reservoir structures are obtained by connecting (possibly through trained connections) different smaller reservoirs constructed in a randomised manner.

Our CRJ reservoirs can also be related to the work of (Deng and Zhang, 2007) where massive reservoirs are constructed in a randomised manner so that they exhibit small-world and scale-free properties of complex networks. We refer to this model as the small world network reservoir (SWNR). We trained such SWNR architecture on the laser and NARMA datasets, since for reasonable results the SWNR model needed to be of larger

size, we conducted the comparative experiments with reservoirs of size $N = 500$. The results (across 10 randomised SWNR model construction runs) for laser and NARMA data sets are presented in Table 4.7 . The performance was always inferior to our simple deterministically constructed CRJ reservoir. Detailed parameter settings of the selected model representatives can be found in Appendix B table B.2.

Finally, we mention that in the context of the work presented in this chapter, the work done in the complex network community, relating dynamics of large networks with different degrees of constrained interconnection topology between nodes, may be of interest. For example, Watts and Strogatz (1998) consider collective dynamics of networks with interconnection structure controlled from completely regular (each node on a ring connects to its k nearest neighbours), through "small-world" (for each node, with some probability p links to the nearest neighbours are rewired to any randomly chosen node on the ring), to completely random (p=1). However, such studies address different issues from those we are concerned with in this work: first, our reservoirs are input-driven; second, our interconnection construction is completely deterministic and regular; and third, the dynamics of CRJ is given through affine functions in every node, put through a saturation sigmoid-type activation functions.

Table 4.7: Test Set NMSE Results of ESN, SWNR, Deterministic SCR and Deterministic CRJ reservoir Model on the Santa Fe Laser Dataset and NARMA System. Reservoir Size $N = 500$ and Reservoir Nodes with *tanh* Transfer Function were Used.

Dataset	ESN	SWNR	SCR	CRJ
laser	0.00724 (0.00278)	0.00551 (0.00176)	0.00816	0.00512
NARMA	0.0104 (0.0020)	0.052 (0.0089)	0.0216	0.0081

4.4 Reservoir Characterisations

There has been a stream of research work trying to find useful characterisations of reservoirs that would correlate well with the reservoir performance on a number of tasks. For example, (Legenstein and Maass, 2007) introduce a 'kernel' measure of separability of different reservoir states requiring different output values. Since linear readouts are used, the separability measure can be calculated based on the rank of the reservoir design matrix (reservoir states resulting from driving the reservoir with different input streams). In the same vein, Bertschinger and Natschlager (2004) suggested that if a reservoir model is to be useful for computations on input time-series, it should have the "separation property" - different input time series which produce different outputs should have different reservoir representations. When linear readouts are used, this typically translates to 'significantly' different states. Moreover, it is desirable that the separation (distance between reservoir states) increases with the difference of the input signals.

In what follows we examine three other reservoir characterisations suggested in the literature, namely - eigenspectrum of the reservoir weight matrix (Ozturk et al., 2007), (pseudo) Lyapunov exponent of the input-driven reservoir dynamics (Verstraeten et al., 2007) and short term memory capacity (Jaeger, 2002b).

4.4.1 EigenSpectra of Dynamic Reservoirs

Several studies have attempted to link eigenvalue distribution of the ESN reservoir matrix W with the reservoir model's performance. First, in order to account for echo state property, the eigenvalues of W need to lie inside the unit circle. Ozturk, Xu and Principe (Ozturk et al., 2007) proposed that the distribution of reservoir activations should have high entropy. It is suggested that the linearised ESN designed with the recurrent weight matrix having the eigenvalues uniformly distributed inside the unit circle creates such an activation distribution (compared to other ESNs with random internal connection

weight matrices). In such cases, the system dynamics will include uniform coverage of time constants (related to the uniform distribution of the poles) (Ozturk et al., 2007). However, empirical comparison of this type of reservoir with the standard ESN is still lacking (Lukosevicius and Jaeger, 2009).

It has been also suggested that sparsity of reservoir interconnections (non-zero entries in W) is a desirable property (Jaeger and Hass, 2004). On the other hand, (Zhang and Wang, 2008) argue that sparsely and fully connected reservoirs in ESN have the same limit eigenvalue distribution inside the unit circle. Furthermore, the requirement that the reservoir weight matrix be scaled so that the eigenvalues of W lie inside the unit circle has been criticised in (Verstraeten et al., 2006), where the experiments show that scaling with a large spectral radius seemed to be required for some tasks. On the other hand, smaller eigenvalue spread is necessarily for stable online training of the readout (Jaeger, 2005).

Our experimental results show that the simple CRJ and regular hierarchical CRHJ reservoirs outperform standard randomised ESN models on a wide variety of tasks. However, the eigenvalue spectra of our regularly and deterministically constructed reservoirs are much more constrained than those of the standard ESN models. Figure 4.6 shows the eigenvalue distribution of representatives of the four model classes - ESN, SCR, CRJ, and CRHJ - fitted on the isolated digits dataset in the speech recognition task. Clearly the coverage of the unit circle by the ESN eigenvalues is much greater than in the case of the three regular deterministic reservoir constructions. While the ESN eigenvalues cover the unit sphere 'almost uniformly', the SCR, CRJ, and CRHJ eigenvalues are limited to a circular structure inside the unit disk. The eigenvalues of SCR must lie on a circle by definition. On the other hand, the eigenvalue structure of CRJ and CRHJ can be more varied. However, the eigenvalue distributions of CRJ and CRHJ reservoirs selected on datasets used in this study were all highly constrained following an approximately circular structure. This poses a question as to what aspects of eigenvalue distribution of

the reservoir matrix are relevant for a particular class of problems. We suspect that the non-linear nature of the non-autonomous reservoir dynamics may be a stumbling block in our efforts to link linearised autonomous behaviour of reservoirs with their modelling potential as non-linear non-autonomous systems.

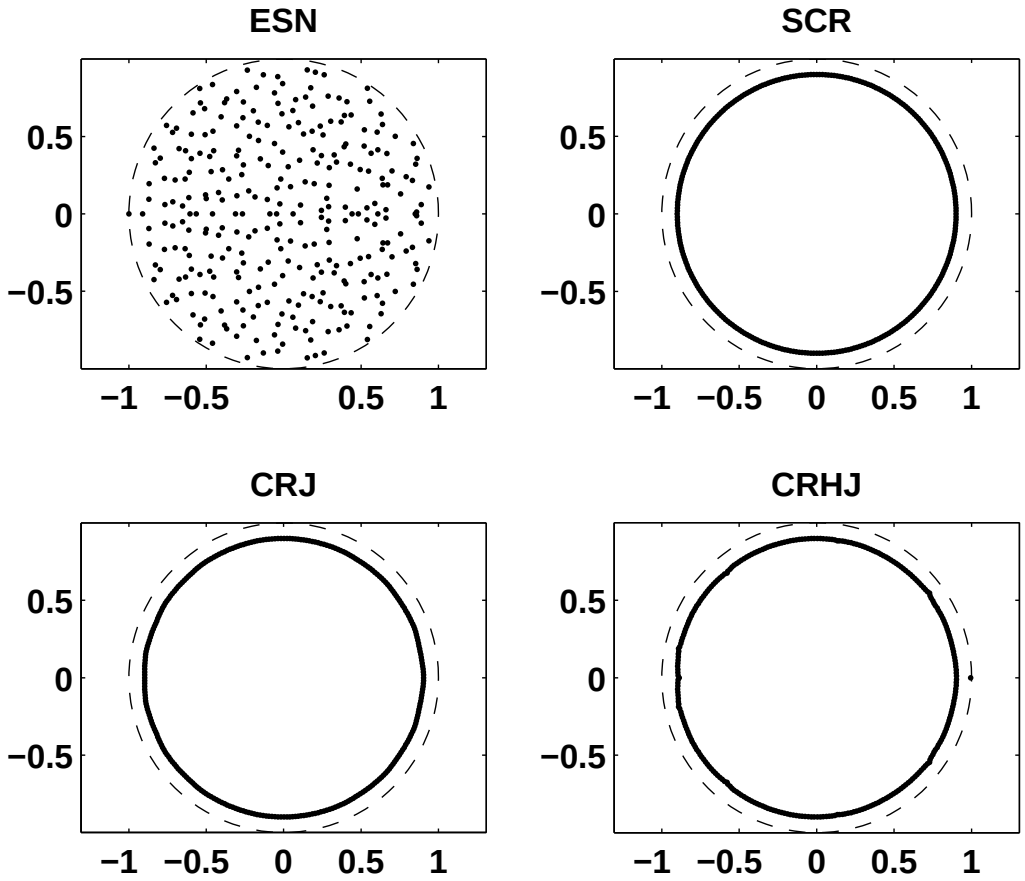

Figure 4.6: Eigenvalue Distribution for ESN, SCR, CRJ and CRHJ Reservoirs of $N = 300$ Neurons Selected on the Isolated Digits Dataset in the Speech Recognition Task (and Hence Used to Report Results in Table 4.4).

4.4.2 Memory Capacity

Another attempt at characterisation of dynamic reservoirs is in terms of their (short-term) memory capacity (MC) (Jaeger, 2002a). It quantifies the ability of recurrent network architectures to encode past events in their state space so that past items in an i.i.d.

input stream can be recovered (at least to certain degree).
Consider a univariate stationary input signal $s(t)$ driving the network at the input layer. For a given delay k, we construct a network with optimal parameters for the task of outputting $s(t-k)$ after seeing the input stream $...s(t-1)s(t)$ up to time t. The goodness of fit is measured in terms of the squared correlation coefficient between the desired output (input signal delayed by k time steps) and the observed network output $y(t)$ see eq.(2.16), and the short term memory (STM) capacity is then given by eq.(2.17).

Traditionally, memory capacity has been estimated numerically by generating long input streams of i.i.d data and training different readouts for different delays k from 1 up to some upper bound k_{max}. Typically, due to short-term memory of reservoir models, k_{max} is of order 10^2. We will later show that such empirical estimations of MC_k, even for linear reservoirs, are inaccurate, especially for larger values of k.

Jaeger (2002a) proved that for *any* recurrent neural network with N recurrent neurons, under the assumption of i.i.d. input stream, MC cannot exceed N. We proved in section 3.3 (under the assumption of zero-mean i.i.d. input stream) that MC of linear SCR architecture with N reservoir units can be made arbitrarily close to N. In particular, $MC = N - (1 - r^{2N})$, where $r \in (0, 1)$ is the single weight value for all connections in the cyclic reservoir. In order to study the memory capacity structure of linear SCR and the influence of additional shortcuts in CRJ, we first present a novel way of estimation of MC_k directly from the reservoir matrix.

4.4.2.1 Direct Memory Capacity Estimation for Linear Reservoirs

Given a (one side infinite) i.i.d. zero-mean real-valued input stream $s(..t) = ... \; s(t-3) \; s(t-2) \; s(t-1) \; s(t)$ emitted by a source P, the state (at time t) of the linear reservoir

with reservoir weight matrix W and input vector V is

$$x(t) = \sum_{\ell=0}^{\infty} s(t-\ell)\ W^{\ell}\ V$$

For the task of recalling the input from k time steps back, the optimal least-squares readout vector U is given by eq.(3.9).

Then the covariance matrix can be evaluated as

$$\begin{aligned} R &= E_{P(s(..t))}\left[\left(\sum_{\ell=0}^{\infty} s(t-\ell)\ W^{\ell}\ V\right)\cdot\left(\sum_{q=0}^{\infty} s(t-q)\ W^{q}\ V\right)^{T}\right] \\ &= E_{P(s(..t))}\left[\sum_{\ell,q=0}^{\infty} s(t-\ell)\ s(t-q)\ W^{\ell}\ VV^{T}\ (W^{q})^{T}\right] \\ &= \sum_{\ell,q=0}^{\infty} E_{P(s(..t))}[s(t-\ell)\ s(t-q)]\ W^{\ell}\ VV^{T}\ (W^{T})^{q} \\ &= \sigma^{2}\sum_{\ell=0}^{\infty} W^{\ell}\ VV^{T}\ (W^{T})^{\ell}, \end{aligned} \tag{4.2}$$

where σ^2 is the variance of the i.i.d. input stream.

Analogously,

$$\begin{aligned} p^{(k)} &= E_{P(s(..t))}\left[\sum_{\ell=0}^{\infty} s(t-\ell)\ s(t-k)\ W^{\ell}\ V\right] \\ &= \sum_{\ell=0}^{\infty} E_{P(s(..t))}[s(t-\ell)\ s(t-k)]\ W^{\ell}\ V \\ &= \sigma^{2}\ W^{k}\ V. \end{aligned} \tag{4.3}$$

Provided R is full rank, by (3.9), (4.2) and (4.3), the optimal readout vector $U^{(k)}$ for delay $k \geq 1$ reads

$$U^{(k)} = G^{-1}\ W^{k}\ V, \tag{4.4}$$

where

$$G = \sum_{\ell=0}^{\infty} W^{\ell}\ VV^{T}\ (W^{T})^{\ell}. \tag{4.5}$$

The optimal 'recall' output at time t is then

$$\begin{aligned} y(t) &= x^{T}(t)\ U^{(k)} \\ &= \sum_{\ell=0}^{\infty} s(t-\ell)\ V^{T}\ (W^{\ell})^{T}\ G^{-1}\ W^{k}\ V, \end{aligned} \tag{4.6}$$

yielding

$$\begin{aligned} Cov(s(t-k), y(t)) &= \sum_{\ell=0}^{\infty} E_{P(s(..t))}[s(t-\ell)\ s(t-k)]\ V^{T}\ (W^{\ell})^{T}\ G^{-1}\ W^{k}\ V \\ &= \sigma^{2}\ V^{T}\ (W^{k})^{T}\ G^{-1}\ W^{k}\ V. \end{aligned} \tag{4.7}$$

Since for the optimal recall output $Cov(s(t-k), y(t)) = Var(y(t))$ by eq.(3.16), we have

$$MC_{k} = V^{T}\ (W^{k})^{T}\ G^{-1}\ W^{k}\ V. \tag{4.8}$$

Two observations can be made at this point. First, as proved by Jaeger (2002a), MC_k constitute a decreasing sequence in $k \geq 1$. From (4.8) it is clear that MC_k scale as $\|W\|^{2k}$, where $\|W\| < 1$ is a matrix norm of W. Second, denote the image of the input weight vector V through k-fold application of the reservoir operator W by $V^{(k)}$, i.e. $V^{(k)} = W^{k}\ V$. Then the matrix $G = \sum_{\ell=0}^{\infty} V^{(\ell)}\ (V^{(\ell)})^{T}$ can be considered a scaled 'covariance' matrix of the iterated images of V under the reservoir mapping. In this interpretation, MC_k is nothing but the squared 'Mahalanobis norm' of $V^{(k)}$ under such covariance structure,

$$\begin{aligned} MC_{k} &= (V^{(k)})^{T}\ G^{-1}\ V^{(k)} \\ &= \|V^{(k)}\|^{2}_{G^{-1}}. \end{aligned} \tag{4.9}$$

We will use the derived expressions to approximate the memory capacity of different kinds of (linear) reservoirs to a much greater degree of precision than that obtained through the usual empirical application of the definition in (2.16) - first generate a long series of i.i.d. inputs and drive with it the reservoir; then train the readout to recover the inputs delayed by k time steps; finish by numerically estimating the statistical moments in (2.16) using the target values (delayed inputs) and their estimates provided at ESN output.

We will approximate $G = \sum_{\ell=0}^{\infty} V^{(\ell)} \ (V^{(\ell)})^T$ by a finite expansion of the first L terms

$$\hat{G}(L) = \sum_{\ell=0}^{L} V^{(\ell)} \ (V^{(\ell)})^T. \tag{4.10}$$

We have

$$\begin{aligned} \|V^{(\ell)}\|_2 &\leq \|W^\ell\|_F \cdot \|V\|_2 \\ &\leq \sqrt{N} \cdot \|W^\ell\|_2 \cdot \|V\|_2 \\ &\leq \sqrt{N} \cdot \|W\|_2^\ell \cdot \|V\|_2 \\ &= \sqrt{N} \cdot (\sigma_{max}(W))^\ell \cdot \|V\|_2, \end{aligned} \tag{4.11}$$

where $\|\cdot\|_2$ and $\|\cdot\|_F$ is the (induced) L_2 and Frobenius norm, respectively, and $\sigma_{max}(W)$ is the largest singular value of W. Furthermore,

$$\begin{aligned} \|V^{(\ell)} \ (V^{(\ell)})^T\|_2 &= \|V^{(\ell)}\|_2^2 \\ &\leq N \cdot (\sigma_{max}(W))^{2\ell} \cdot \|V\|_2^2, \end{aligned}$$

and so, given a small $\epsilon > 0$, we can solve for the number of terms $L(\epsilon)$ in the approximation (4.10) of G so that the norm of contributions $V^{(\ell)} \ (V^{(\ell)})^T$, $\ell > L(\epsilon)$, is less than ϵ. Since

$\sigma_{max}(W) < 1$,

$$\begin{aligned} \| \sum_{\ell=L(\epsilon)}^{\infty} V^{(\ell)} \ (V^{(\ell)})^T \|_2 &\leq \sum_{\ell=L(\epsilon)}^{\infty} \|V^{(\ell)} \ (V^{(\ell)})^T\|_2 \\ &\leq N \ \|V\|_2^2 \sum_{\ell=L(\epsilon)}^{\infty} (\sigma_{max}(W))^{2\ell} \\ &= N \ \|V\|_2^2 \ \frac{(\sigma_{max}(W))^{2L(\epsilon)}}{1-(\sigma_{max}(W))^2}, \end{aligned} \tag{4.12}$$

we have that for

$$L(\epsilon) > \frac{1}{2} \ \frac{\log \frac{\epsilon \ (1-(\sigma_{max}(W))^2)}{N \ \|V\|_2^2}}{\log \sigma_{max}(W))}, \tag{4.13}$$

it holds

$$\| \sum_{\ell=L(\epsilon)}^{\infty} V^{(\ell)} \ (V^{(\ell)})^T \|_2 \leq \epsilon,$$

and so with $L(\epsilon)$ terms in (4.10), G can be approximated in norm up to a term $< \epsilon$.

4.4.2.2 The Effect of Shortcuts in CRJ on Memory Capacity

In section 3.3 we proved that the 'k-step recall' memory capacity MC_k for the SCR with reservoir weight $r \in (0,1)$ is equal to

$$MC_k = r^{2k} \ (1 - r^{2N}) \ \zeta_{k \bmod N},$$

where $\zeta_j = r^{-2j}$, $j = 0, 1, 2, ..., N-1$. It follows that for $k \geq 1$,

$$\begin{aligned} MC_k &= r^{2k} \ (1 - r^{2N}) \ r^{-2 \ (k \bmod N)} \\ &= (1 - r^{2N}) \ r^{2 \ [k - (k \bmod N)]} \\ &= (1 - r^{2N}) \ r^{2N \ (k \text{ div } N)}, \end{aligned} \tag{4.14}$$

where div represents integer division. Hence, for linear cyclic reservoirs with reservoir weight $0 < r < 1$, MC_k is a non-increasing piecewise constant function of k, with blocks

of constant value

$$MC_{qN+j} = (1 - r^{2N})\, r^{2Nq}, \quad q \geq 0, \quad j \in \{0, 1, ..., N-1\}. \tag{4.15}$$

In order to study the effect of reservoir topologies on the contributions MC_k to the memory capacity MC, we first selected three model class representatives (on the validation set) with $N = 50$ linear unit reservoirs on the system identification task (10-th order NARMA), one representative for each of the model classes ESN, SCR and CRJ (jump length 4). Linear and non-linear reservoirs of size 50 had similar performance levels on the NARMA task. To make the MC_k plots directly comparable, we then re-scaled the reservoir matrices W to a common spectral radius $\rho \in (0, 1)$. In other words, we are interested in differences in the profile of MC_k for different reservoir types, as k varies. Of course, for smaller spectral radii, the MC contributions will be smaller, but the principal differences can be unveiled only if the same spectral radius is imposed on all reservoir structures.

The memory capacity of the reservoir models was estimated through estimation of MC_k, $k = 1, 2, ..., 200$, in two ways:

1. *Empirical Estimation:* The i.i.d. input stream consisted of 9000 values sampled from the uniform distribution on $[-0.5, 0.5]$. The first 4000 values were used for training, the next 2000 for validation (setting the regularisation parameter of Ridge regression in readout training), and the remaining 3000 values was used for testing the models (prediction of the delayed input values). After obtaining the test outputs, the memory capacity contributions MC_k were estimated according to (2.16). This process was repeated 10 times (10 runs), in each run a new input series has been generated. Final MC_k estimates were obtained as averages of the MC_k estimated across the 10 runs. This represents the standard approach to MC estimation proposed by Jaeger (2002a) and used in the ESN literature (Fette and Eggert, 2005;

Ozturk et al., 2007; Verstraeten et al., 2007; Steil, 2007).

2. *Theoretical Estimation:* The MC contributions MC_k were calculated from (4.8), with G approximated as in (4.10). The number of terms L has been determined according to (4.13), where the precision parameter ϵ was set to $\epsilon = 10^{-60}$.

Figures 4.7(A) and (B) present theoretical and empirical estimates, respectively, of MC_k for $\rho = 0.8$. Analogously, Figures 4.7(C) and (D) show theoretical and empirical estimates of MC_k for $\rho = 0.9$. The direct theoretical estimation (Figures 4.7(A,C)) is much more precise than the empirical estimates (Figures 4.7(B,D)). Note the clear step-wise behaviour of MC_k for SCR predicted by the theory (eq. (4.15)). As predicted, the step size is $N = 50$. In contrast, the empirical estimations of MC_k can infer the first step at $k = 50$, but lack precision thereafter (for $k > 50$). Interestingly, SCR topology can keep information about the last $N-1$ i.i.d. inputs to a high level of precision ($MC_k = 1-r^{2N}$, $k = 1, 2, ..., N-1$), but then loses the capacity to memorise inputs more distant in the past in a discontinuous manner (jump at $k = N = 50$). This behaviour of MC_k for SCR is described analytically by eq. (4.15). In contrast, as a consequence of 'cross-talk' effects introduced by jumps in CRJ, the MC contributions MC_k start to rapidly decrease earlier than at $k = N$, but the reservoir can keep the information about some of the later inputs better than in the case of SCR (roughly for $50 \leq k \leq 60$). In the case of ESN, the MC_k values decrease more rapidly than in the case of both SCR and CRJ. Using the standard empirical estimation of MC_k, such a detailed behaviour of memory capacity contributions would not be detectable. To demonstrate the potential of our method, we show in Figures 4.8(A,B) theoretically determined graphs of MC_k for delays up to $k = 400$ using $\rho = 0.8$ (A) and $\rho = 0.9$ (B).

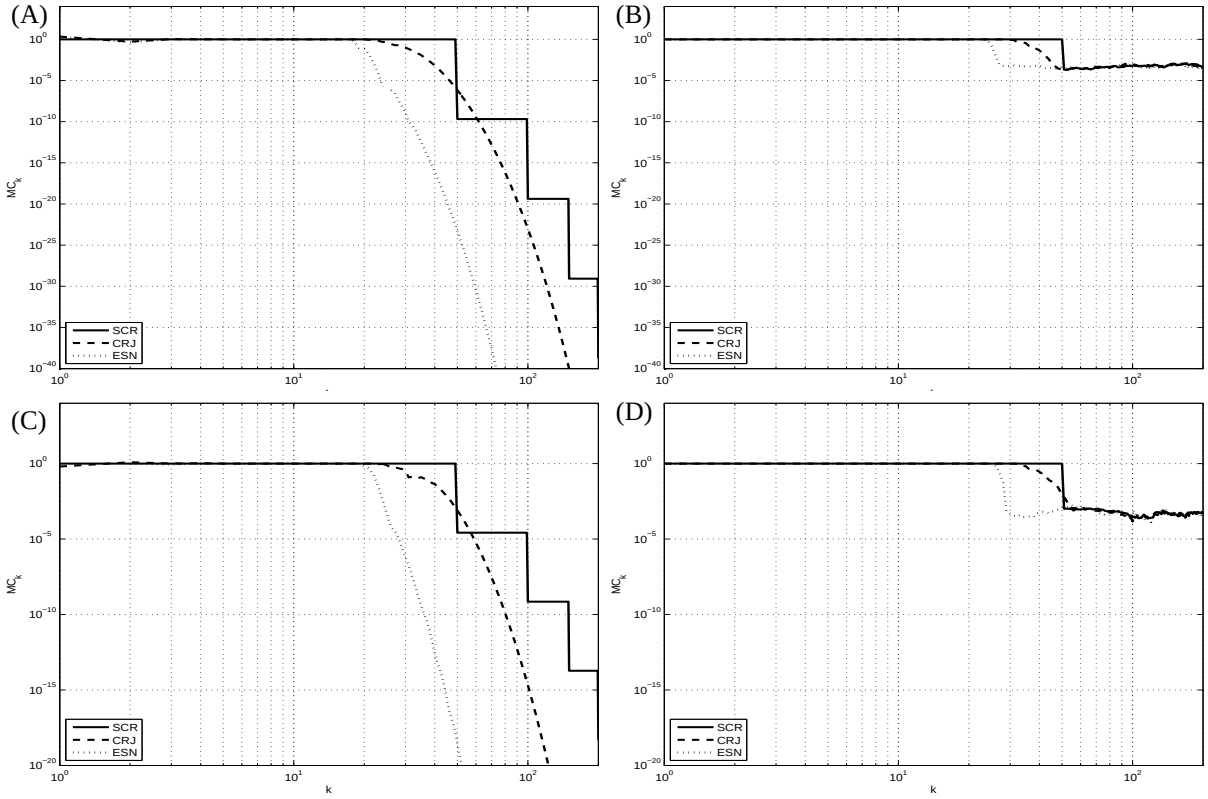

Figure 4.7: Theoretical (A,C) and Empirical (B,D) k-Delay MC of ESN (dotted line), SCR (solid line), and CRJ (dashed line) for Delays $k = 1, ..., 200$. The Graphs of MC_k are shown for $\rho = 0.8$ (A,B) and $\rho = 0.9$ (C,D).

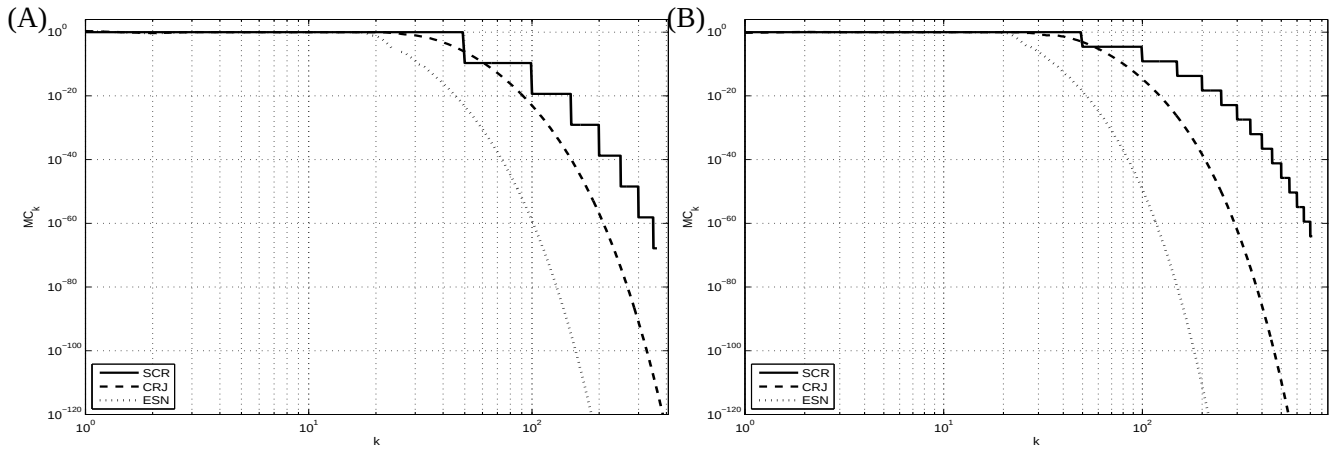

Figure 4.8: Theoretical k-Delay MC of ESN (dotted line), SCR (solid line), and CRJ (dashed line) for Delays $k = 1, ..., 400$. The Graphs of MC_k are shown for $\rho = 0.8$ (A) and $\rho = 0.9$ (B).

4.4.3 Lyapunov Exponent

Verstraeten et al. (2007) suggest to extend numerical calculation of the well known Lyapunov exponent characterisation of (ergodic) autonomous dynamical systems to input-driven systems. The same idea occurred previously in the context of recurrent neural networks for processing symbolic streams (Tabor, 2002). While the reservoir is driven by a particular input sequence, at each time step the local dynamics is linearised around the current state and the Lyapunov spectrum is calculated. In our experiments the selected ESN configurations in the laser, NARMA and speech recognition tasks all led to pseudo-Lyapunov exponents ranging from 0.35 to 0.5. As in Verstraeten et al. (2007), the found exponents are positive, suggesting local exponential divergence along the sampled reservoir trajectories, and hence locally expanding systems (at least in one direction). For our simple reservoir architectures, SCR and CRJ, the selected configurations across the data sets also lead to similar pseudo-Lyapunov exponents, but this time in the negative range. For example the CRJ exponents ranged from -0.4 to -0.25. All exponents for the selected architectures of both SCR and CRJ were negative, implying contractive dynamics.

To study the pseudo-Lyapunov exponents of the selected reservoir architectures along the lines of (Verstraeten et al., 2007), for each data set, the reservoir matrix of each selected model representative from ESN, SCR and CRJ was rescaled so that the spectral radius ranged from 0.1 to 2. The resulting pseudo-Lyapunov exponents are shown in Figure 4.9 for the NARMA (A), laser (B), and speech (C) data sets. The vertical lines denote the spectral radii of the selected 'optimal' model representatives and black markers show the corresponding exponents. Interestingly, for all data sets, the pseudo-Lyapunov exponent lines of ESN are consistently above the SCR ones, which in turn are above those of CRJ. This ranking holds also for the selected model representatives on different tasks. Our results show that a reservoir model can have superior performance without expanding dynamics. In fact, in our experiments the CRJ reservoir achieved the best results while having on average contractive dynamics along the sampled trajectories and

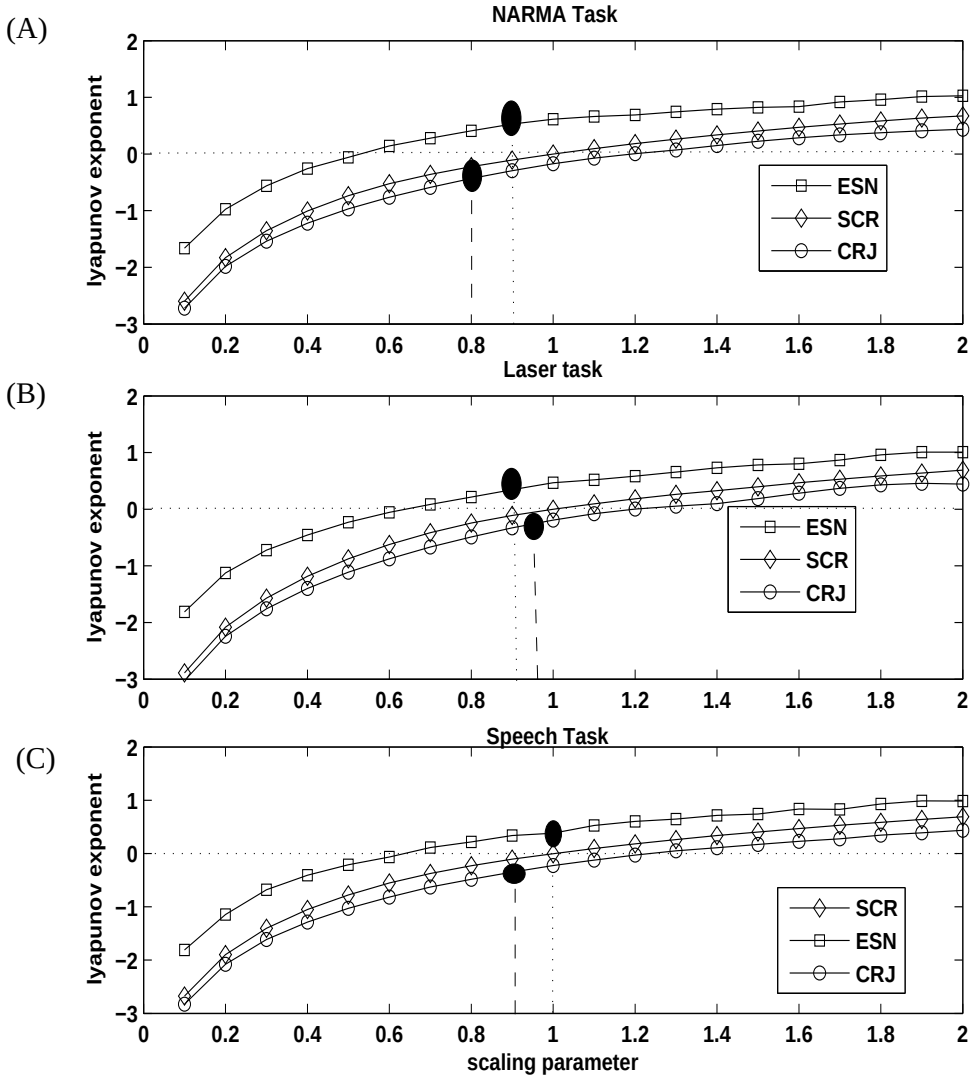

Figure 4.9: Pseudo-Lyapunov Exponents for ESN, SCR, and CRJ on the NARMA (A), Laser (B), and Speech Recognition (C) Tasks. The Vertical Lines Denote the Spectral Radii of the Selected 'Optimal' Model Representatives and Black Markers Show the Corresponding Exponents.

the least pseudo-Lyapunov exponent.

4.5 Chapter Summary

In this chapter, first, we have introduced a novel simple deterministic reservoir model, Cycle Reservoir with Jumps (CRJ, see section 4.1), that has superior performance to standard ESN on a variety of temporal tasks of different origin and characteristics (see section 4.2). We also investigated three reservoir characterisations (eigen-spectrum of the reservoir weight matrix, short term memory capacity and pseudo-Lyapunov exponent) in the context of reservoir models studied in this work. In section 4.4.1 we showed that for a superior model performance it is not necessary to have a uniform coverage of eigenvalues on the unit disk. Despite having highly constrained eigenvalue distribution, the CRJ consistently outperforms ESN that has more uniform eigenvalue coverage of the unit disk.

Furthermore, in section 4.4.2 we presented a new framework for determining short term memory capacity of linear reservoir models to a high degree of precision. Using this framework, we studied the effect of shortcut (jumps) connections in the CRJ reservoir topology on its memory capacity. Due to cross-talk effects introduced by the jumps in CRJ, the MC contributions start to rapidly decrease earlier than in the case of SCR, but unlike in SCR, the decrease in MC_k in CRJ is gradual, enabling the reservoir to keep more information about some of the later inputs.

Finally, unlike in the case of ESN, pseudo-Lyapunov exponents of the selected 'optimal' CRJ models are consistently negative (see Section 4.4.3).

Chapter 5

Negatively Correlated Echo State Networks

In this chapter we apply the idea of Negative Correlation learning (NCL) to the ensemble of Echo State Networks (ESNs). Each ESN operates with a different reservoir, possibly capturing different features of the input stream. On each reservoir we build a non-linear readout mapping. Crucially, the individual readouts of the ensemble are coupled together by a diversity-enforcing term of the NCL training, which may have a potential to stabilise the overall collective ensemble output. The chapter is organised as follows. Section 5.1 presents our model, Ensemble of ESNs using NCL. Experiments and Results are presented and discussed in Sections 5.2 and 5.3, respectively. Finally, this chapter is summarised in section 5.4

5.1 Ensembles of ESNs using NCL

Negative Correlation Learning (NCL) has been successfully applied to training MLP ensembles (Brown et al., 2005.; Brown and Yao, 2001; Liu and Yao, 1999; Mckay and Abbass,

2001). In NCL, all the individual networks are trained simultaneously and interactively through the correlation penalty terms in their error functions.

To apply NCL to ensembles of ESN, we replace the linear readouts of individual standard ESN with non-linear Multi-Layer Perceptron (MLP). To exploit the power of negative correlation the ensemble members should be non-linear models. Negatively correlated linear mappings cannot implement the idea of globally correct mappings by all ensemble members, while being locally diverse.

The training of negatively correlated ensemble of M ESNs consists of first, driving the individual ESN reservoirs with the input stream and collecting the reservoir states $x^i(t) = (x^i_1(t).....x^i_N(t))$, where $x^i(t)$ is the reservoir activation vector of the i-th ESN, $i = 1, 2, ..., M$, at time t. Each ESN i has N reservoir units with reservoir weight matrix W^i and input matrix V^i.

Each reservoir state can be updated and collecting according to:

$$x^i(t) = f(V^i s(t) + W^i x^i(t-1)), \tag{5.1}$$

where f is the reservoir activation function (tanh in this study).

We then use the reservoir states $x^i(t)$ as an input for the MLP readouts F_i (see figure 5.1). The readout is computed as:

$$F_i(x^i(t)) = g(x^i(t)), \tag{5.2}$$

where g is the non-linear MLP readout function. The readout mapping can be trained in an offline or online mode by minimising the Mean Square Error,

$$MSE = \langle (F_i(x^i(t)) - y(t))^2 \rangle, \tag{5.3}$$

where $F_i(x^i(t))$ is the readout output of the i-th MLP, $y(t)$ is the desired output (target),

and $\langle\cdot\rangle$ denotes the empirical mean. The readout MLPs had a single hidden layer of logistic sigmoid units (the hidden layer size was determined through cross-validation) and were trained using NCL.

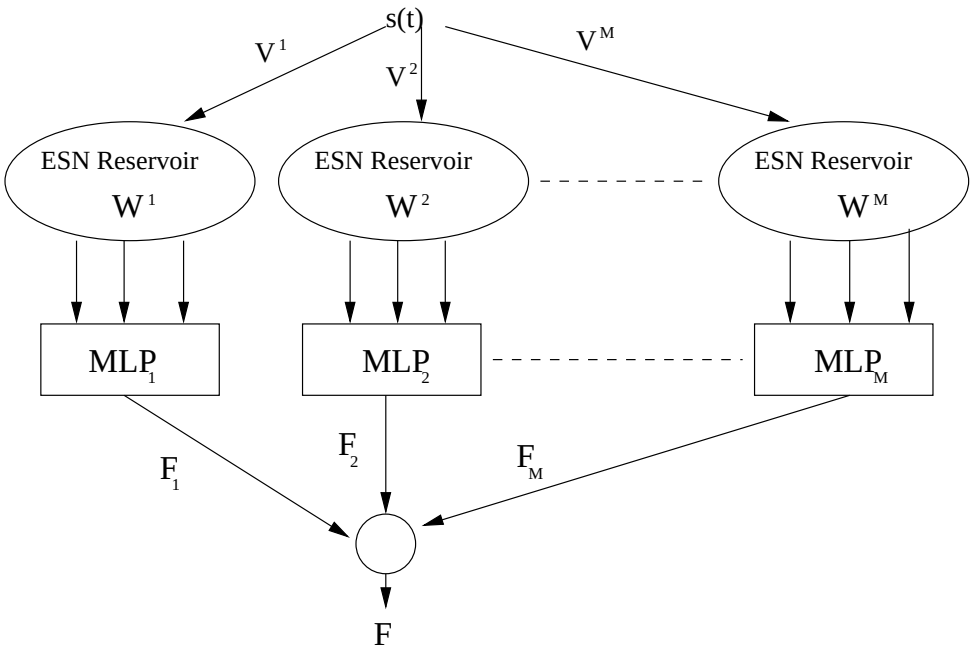

Figure 5.1: Ensemble of ESN with MLP readouts.

The ensemble output $F(t)$ is calculated as a flat average over all ensemble members $F_i(x(t))$,

$$F(t) = \frac{1}{M}\sum_{i=1}^{M}(F_i(x^i(t))). \tag{5.4}$$

In NCL the penalised error functional to be minimised reads:

$$E_i = \frac{1}{2}(F_i(x^i(t)) - y(t))^2 + \lambda p_i(x^i(t)), \tag{5.5}$$

where

$$p_i(x^i(t)) = (F_i(x^i(t)) - F(t))\sum_{i\neq j}(F_j(x^j(t)) - F(t)), \tag{5.6}$$

and $\lambda > 0$ is an adjustable strength parameter for the negative correlation enforcing penalty term p_i. It can be shown that:

$$E_i = \frac{1}{2}(F_i(x^i(t)) - y(t))^2 - \lambda(F_i(x^i(t)) - F(t))^2. \tag{5.7}$$

Note that when $\lambda = 0$, we obtain a standard de-coupled training of individual ensemble members. Standard gradient-based approaches can be used to minimise E by updating the parameters of each individual ensemble member.

We remark that in contrast to standard NCL, in ensemble of ESNs, the maps F_i each receive a different input $x^i(t)$ that provide diverse representations of the common input stream $...s(t-1)s(t)$ observed up to time t. However, one can treat the reservoir activations $x^i(t)$ as internal representations of the i-th ensemble model receiving the common input $s(t)$. From this point of view, all the ensemble models receive the same input, as is the case in the standard NCL.

5.2 Experiments

We employ three timeseries used in the Echo state Network (ESN) literature and introduced in section 3.2.1 to evaluate our proposed Ensemble of ESN, 10th order NARMA system (Verstraeten et al., 2007), Laser Dataset (Steil, 2007), and Sunspot series (Schwenker and Labib, 2009). For each data set, we denote the length of the training, validation and test sequences by L_{trn}, L_{val} and L_{tst}, respectively. The first L_{wash} values from training, validation and test sequences are used as the initial washout period. In what follows we briefly introduce the data sets.

5.2.1 Datasets

10th order NARMA system

$$y(t+1) = 0.3\ y(t) + 0.05\ y(t) \sum_{i=0}^{9} y(t-i) + 1.5\ s(t-9)\ s(t) + 0.1, \tag{5.8}$$

The networks were trained on system identification task to output $y(t)$ based on $s(t)$, with $L_{trn} = 2000$, $L_{val} = 3000$, $L_{tst} = 3000$ and $L_{wash} = 200$.

Chaotic Laser Dataset

The time series is a cross-cut through periodic to chaotic intensity pulsations of a real laser. The task is to predict the next laser activation $y(t+1)$, given the values up to time t; $L_{trn} = 2000$, $L_{val} = 3000$, $L_{tst} = 3000$ and $L_{wash} = 200$.

Sunspot series

This dataset contains 3100 sunspots numbers from Jan 1749 to April 2007, where $L_{trn} = 1600$, $L_{val} = 500$, $L_{tst} = 1000$ and $L_{wash} = 100$. The task was to predict the next value $y(t+1)$ based on the history of y up to time t.

5.2.2 Experimental setup

The ensemble used in our experiments consists of $M = 10$ ESNs with MLP readouts. In all experiments we use ESNs with reservoirs of $N = 100$ units. Hence, each individual MLP readout has 100 inputs. We used NCL training of readouts via gradient descent on E with learning rate $\eta = 0.1$. The output activation function of the MLP readout was linear for NARMA task and sigmoid logistic for the laser and sunspot tasks.

We optimised the penalty factor λ and the readout complexity (number of hidden nodes in F_i) using the validation set, λ was varied in the range $[0, 1]$ (step size 0.1) (Brown and Yao, 2001). The number of hidden nodes was varied from 1 to 20 (step 1).

The single ESN model architecture described by hyperparameters such as input weight scale, spectral radius and reservoir sparsity, was determined on the validation set. Linear readout was trained via ridge regression (Wyffels et al., 2008).

The performance of this model was determined in 10 independent runs (e.g. 10 realisations of ESN based on the best performing hyperparameters).

For ensemble ESN (Ens-ESN-MLP), we used the 10 ESN reservoirs generated in the single ESN experiment as the ensemble members. Due to random initialisation of MLP readouts, we report the average performance (plus the minimum, maximum and standard deviation values) over 10 random initialisations of MLPs.

5.3 Results

Table 5.1 summarises the results of the single ESN model, Negatively Correlated ensemble of ESNs and independent ensemble of ESNs ($\lambda = 0$) for the three time series considered in this chapter. To assess the improvement achieved by using a genuine NCL training vs. independent training of ensemble members ($\lambda = 0$), the MLP readouts were initialised with the same weight values in both cases. In all datasets, the ESN ensemble trained via NCL outperformed the other models, with the most significant performance gain for NARMA and Sunspots tasks (confidence level 99.9%). For the laser dataset the significance level was greater than 98%.

Note that the two ESN ensemble versions we study share the same number of free parameters, with the sole exception of the single diversity-imposing parameter λ in NCL based learning. The single ESN has been used as a natural baseline against which to

compare the ensemble performance.

Table 5.1: Performance of the single ESN model and the ESN ensemble models

Dataset	Test	ESN	Ens-ESN-MLP	Ens-ESN-MLP
		linear regression	Indep. learning	NCL
NARMA	MSE	0.00102	0.000795	**0.000297**
	STD	0.000101	0.0000142	0.0000237
	Min	0.000865	0.000768	0.000270
	Max	0.00118	0.000810	0.000349
Laser	MSE	0.000197	0.000187	**0.000138**
	STD	0.0000724	0.00000767	0.00000205
	Min	0.0000998	0.000172	0.0000987
	Max	0.000315	0.000197	0.000170
Sunspots	MSE	0.00163	0.00136	**0.00115**
	STD	0.000122	6.385E-06	1.054E-05
	Min	0.00143	0.00136	0.00110
	Max	0.00191	0.00138	0.00116

5.4 Chapter Summary

In this chapter we proposed an ensemble of Echo State Networks (ESNs) with diverse reservoirs whose collective read-out is obtained through Negative Correlation Learning (NCL) of ensemble of Multi-Layer Perceptrons (MLP), where each individual MPL realises the readout from a single ESN. Experimental results on three data sets confirm that, compared with both single ESN and flat ensembles of ESNs, NCL based ESN ensembles achieve better generalisation performance.

Chapter 6

Short Term Memory Quantifications in Input-Driven Linear Dynamical Systems

Input driven dynamical systems play an important role as machine learning models when data sets exhibit temporal dependencies, e.g. in prediction or control. In an attempt to characterise dynamic properties of such systems, measures have been suggested to quantify how well past information can be represented in the system's internal state. In this chapter we investigate two such well known measures, namely the short term memory capacity spectrum MC_k (Jaeger, 2002a) see section 2.2.3, and the Fisher memory curve $J(k)$ (Ganguli et al., 2008). The two quantities map the memory structure of the system under investigation from two quite different perspectives. So far their relation has not been closely investigated. In this work we take the first step to bridge this gap and show that under some conditions MC_k and $J(k)$ can be closely related.

6.1 Fisher Memory Curve (FMC)

Memory capacity MC of a reservoir is one way of quantifying the amount of information that can be preserved in the reservoir about the past inputs. In (Ganguli et al., 2008) Ganguli, Huh and Sompolinsky proposed a different quantification of memory capacity for linear reservoirs corrupted by a Gaussian state noise. In particular, it is assumed that the dynamic noise $z(t)$ is a memoryless process of i.i.d. zero mean Gaussian variables with co-variance ϵI (I is the identity matrix). Then, given an input driving stream $s(..t) = ... \; s(t-2) \; s(t-1) \; s(t)$, the dynamic noise induces a state distribution $p(x(t)|s(..t))$, which is a Gaussian with covariance (Ganguli et al., 2008)

$$C = \epsilon \sum_{\ell=0}^{\infty} W^{\ell} (W^T)^{\ell}. \tag{6.1}$$

The Fisher memory matrix quantifies sensitivity of $p(x(t)|s(..t))$ with respect to small perturbations in the input driving stream $s(..t)$ (parameters of the recurrent network are fixed),

$$F_{k,l}(s(..t)) = -E_{p(x(t)|s(..t))} \left[\frac{\partial^2}{\partial s(t-k) \partial s(t-l)} \log p(x(t)|s(..t)) \right]$$

and its diagonal elements $J(k) = F_{k,k}(s(..t))$ quantify the information that $x(t)$ retain about a change (e.g. a pulse) entering the network k time steps in the past. The collection of terms $\{J(k)\}_{k=0}^{\infty}$ was termed Fisher memory curve (FMC) and evaluated to (Ganguli et al., 2008)

$$J(k) = V^T (W^T)^k C^{-1} W^k V. \tag{6.2}$$

Note that, unlike the short term memory capacity, the FMC does not depend on the input driving stream.

6.2 Relation between short term memory capacity and Fisher memory curve

We first briefly introduce some necessary notation. Denote the image of the input weight vector V through k-fold application of the reservoir operator W by $V^{(k)}$, i.e. $V^{(k)} = W^k\, V$. Define $A = \frac{1}{\epsilon} C - G$, where

$$G = \sum_{\ell=0}^{\infty} V^{(\ell)}\ (V^{(\ell)})^T. \tag{6.3}$$

Provided A is invertible, denote $G\ (A^{-1} + G^{-1})\ G$ by D. For any positive definite matrix $B \in \mathbb{R}^{n\times n}$ we denote the induced norm on $\mathbb{R}^n$ by $\|\cdot\|_B$, i.e. for any $V \in \mathbb{R}^n$, $\|V\|_B^2 = V^T B V$. We are now ready to formulate the main result.

Theorem: *Let MC_k be the k-th memory capacity term (2.16) of network (2.10) with no dynamic noise, under a zero-mean i.i.d. input driving source. Let $J(k)$ be the k-th term of the Fisher memory curve (6.2) of network (2.10) with i.i.d. dynamic noise of variance ϵ. If D is positive definite, then*

$$MC_k = \epsilon\ J(k) + \|V^{(k)}\|_{D^{-1}}^2 \tag{6.4}$$

and $MC_k > \epsilon\ J(k)$, for all $k > 0$.

Proof: Given an i.i.d. zero-mean real-valued input stream $s(..t) = ...\ s(t-2)\ s(t-1)\ s(t)$ of variance σ^2 emitted by a source P, the state at time t of the linear reservoir (under no dynamic noise ($\epsilon = 0$)) is

$$x(t) = \sum_{\ell=0}^{\infty} s(t-\ell)\ W^\ell\ V = \sum_{\ell=0}^{\infty} s(t-\ell)\ V^{(\ell)}.$$

For the task of recalling the input from k time steps back, the optimal least-squares

readout vector U is given by eq.(3.9):

Provided R is full rank, the optimal readout vector $U^{(k)}$ for delay $k \geq 1$ reads

$$U^{(k)} = G^{-1}\ V^{(k)}. \tag{6.5}$$

The optimal 'recall' output at time t is then $y(t) = x^T(t)\ U^{(k)}$, yielding

$$Cov(s(t-k), y(t)) \ = \ \sigma^2\ (V^{(k)})^T\ G^{-1}\ V^{(k)}. \tag{6.6}$$

Since for the optimal recall output $Cov(s(t-k), y(t)) = Var(y(t))$ (Jaeger, 2002a), we have

$$MC_k = (V^{(k)})^T\ G^{-1}\ V^{(k)}. \tag{6.7}$$

The Fisher memory curve and memory capacity terms (6.2) and (6.7), respectively have the same form.

The matrix $G = \sum_{\ell=0}^{\infty} V^{(\ell)}\ (V^{(\ell)})^T$ can be considered a scaled 'covariance' matrix of the iterated images of V under the reservoir mapping. Then MC_k is the squared 'Mahalanobis norm' of $V^{(k)}$ under the covariance structure G,

$$\begin{aligned} MC_k &= (V^{(k)})^T\ G^{-1}\ V^{(k)} \\ &= \|V^{(k)}\|^2_{G^{-1}}. \end{aligned} \tag{6.8}$$

Analogously, $J(k)$ is the squared 'Mahalanobis norm' of $V^{(k)}$ under the covariance C of the state distribution $p(x(t)|s(..t))$ induced by the dynamic noise $z(t)$,

$$\begin{aligned} J(k) &= (V^{(k)})^T\ C^{-1}\ V^{(k)} \\ &= \|V^{(k)}\|^2_{C^{-1}}. \end{aligned} \tag{6.9}$$

Denote the rank-1 matrix VV^T by Q. Then by (6.1),

$$\frac{1}{\epsilon}C = A + G,$$

where

$$A = \sum_{\ell=0}^{\infty} W^{\ell}\ (I - Q)\ (W^T)^{\ell}.$$

It follows that $\epsilon C^{-1} = (A + G)^{-1}$ and, provided A is invertible (and $(A^{-1} + G^{-1})$ is invertible as well), by matrix inversion lemma,

$$\epsilon C^{-1} = G^{-1} - G^{-1}\ (A^{-1} + G^{-1})^{-1}\ G^{-1}.$$

We have

$$\begin{aligned} J(k) &= (V^{(k)})^T\ C^{-1}\ V^{(k)} \\ &= \frac{1}{\epsilon}MC_k - \frac{1}{\epsilon}(V^{(k)})^T\ D^{-1}\ V^{(k)}, \end{aligned}$$

where

$$D = G\ (A^{-1} + G^{-1})\ G.$$

Since G and A are symmetric matrices, so are their inverses and hence D is also a symmetric matrix. Provided D is positive definite, it can be considered (inverse of a) metric tensor and

$$MC_k = \epsilon\ J(k) + \|V^{(k)}\|^2_{D^{-1}}.$$

Obviously, in such a case, $MC_k > \epsilon\ J(k)$ for all $k > 0$.

From (6.4) we have

$$\sum_{k=0}^{\infty} MC_k = \epsilon\ \sum_{k=0}^{\infty} J(k) + \sum_{k=0}^{\infty} \|V^{(k)}\|^2_{D^{-1}}.$$

If the input weight vector V is a unit vector ($\|V\|_2 = 1$) and the reservoir matrix W is normal (i.e. has orthogonal eigenvector basis), we have $\sum_{k=0}^{\infty} J(k) = 1$ (Ganguli et al., 2008). In such cases $\sum_{k=0}^{\infty} MC_k = N$, implying

$$\sum_{k=0}^{\infty} \|V^{(k)}\|^2_{D^{-1}} = N - \epsilon. \tag{6.10}$$

As an example of metric structures underlying the norms in (6.4), (6.8) and (6.9), we show in figure 6.1 covariance structure of C ($\epsilon = 1$), G and D corresponding to a 15-node linear reservoir. The covariances were projected onto the two-dimensional space spanned by the 1st and 14th eigenvectors of C (rank determined by decreasing eigenvalues). Reservoir weights were randomly generated from a uniform distribution over an interval symmetric around zero and then W was normalised to spectral radius 0.995. Input weights were generated from uniform distribution over $[-0.5, 0.5]$.

6.3 Discussion

We investigated the relation between two quantitative measures suggested in the literature to characterise short term memory in input driven dynamical systems, namely the short term memory capacity spectrum MC_k and the Fisher memory curve $J(k)$, for time lags $k \geq 0$. $J(k)$ is independent of the input driving stream $s(..t)$ and measures the 'inherent' memory capabilities of such systems by measuring the sensitivity of the state distribution $p(x(t)|s(..t))$ induced by the dynamic noise with respect to perturbations in $s(..t)$, k time steps back. On the other hand MC_k quantifies how well the past inputs $s(t-k)$ can be reconstructed by linearly projecting the state vector $x(t)$. We have shown, that under some assumptions, the two quantities can be interpreted as squared 'Mahalanobis' norms of images of the input vector under the system's dynamics and that $MC_k > \epsilon\ J(k)$, for all $k > 0$. Even though MC_k and $J(k)$ map the memory structure of the system under

investigation from two quite different perspectives, they can be closely related.

6.4 Chapter Summary

In this chapter we first presented in section 6.1 a review about a quantitative measure suggested in the literature to characterise short term memory in input driven dynamical systems, namely the Fisher memory curve $J(k)$. We have shown in section 6.2, that the short term memory capacity spectrum MC_k and the Fisher memory curve $J(k)$ can be interpreted as squared 'Mahalanobis' norms of images of the input vector. Finally, in section 6.3 we discussed that they can be closely related.

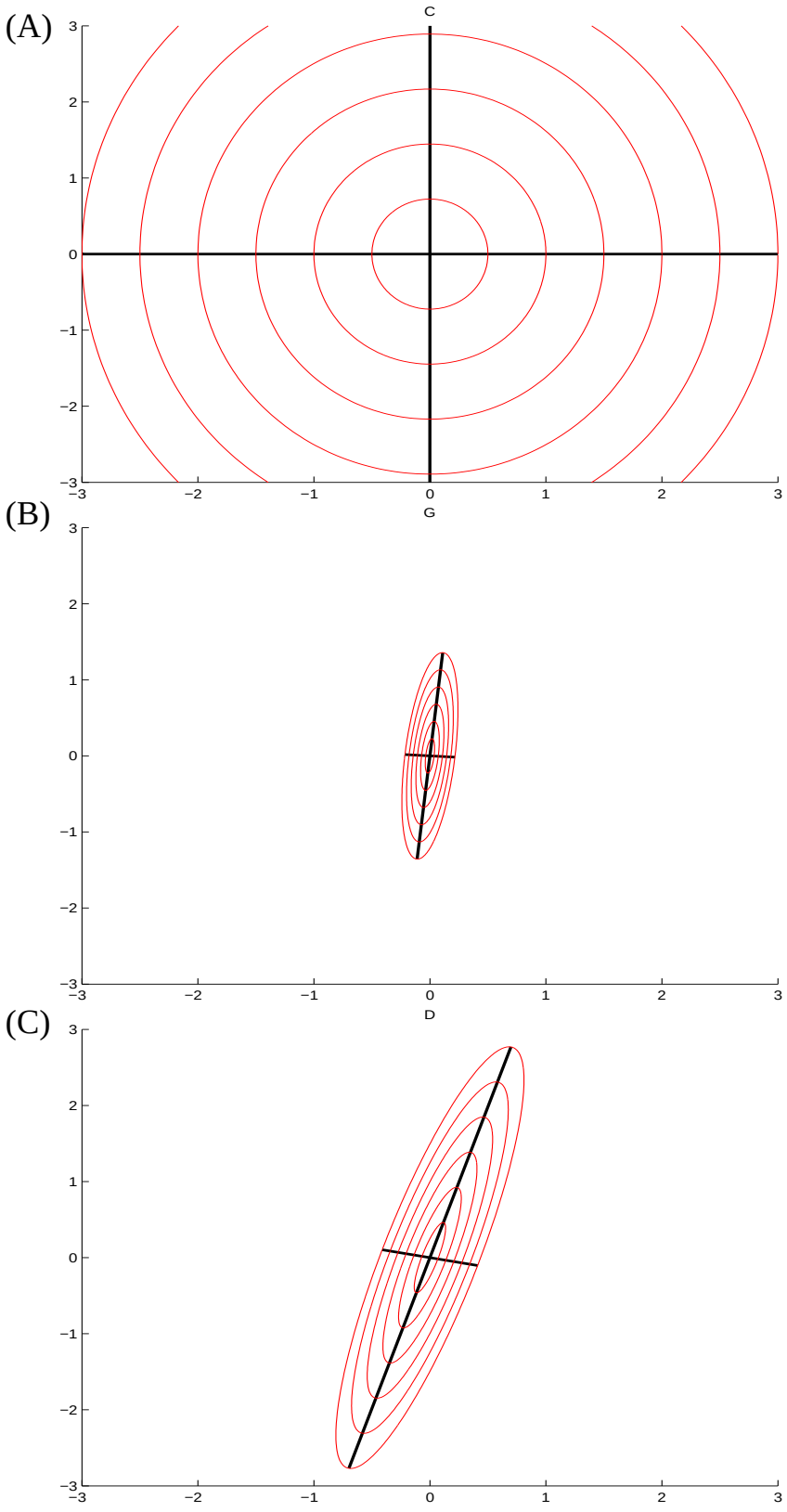

Figure 6.1: Covariance structure of C (A), G (B) and D (C) for a 15-node linear reservoir projected onto the 1st and 14th eigenvectors of C. Shown are iso-lines corresponding to 0.5, 1, 1.5, ..., 3 standard deviations.

Chapter 7

Conclusions and Future Work

This chapter presents the general conclusions and gives directions for future work.

7.1 Conclusions

Reservoir Computing (RC) models are dynamical models for processing time series that make a conceptual separation of the temporal data processing into two parts:

1. representation of temporal structure in the input stream through a non-adaptable dynamic *"reservoir"*, and

2. a memoryless easy-to-adapt *readout* from the reservoir.

The field of reservoir computing has been growing rapidly with dedicated special sessions at conferences and special issues of journals (Jaeger et al., 2007b). It has been widely believed that randomised construction of reservoirs is desirable. Reservoir computing has been successfully applied in many practical applications (Jaeger, 2001, 2002a,b; Jaeger and Hass, 2004; Mass et al., 2004; Tong et al., 2007). However, reservoir computing is sometimes criticised for not being principled enough (Prokhorov, 2005). There

have been several attempts to address the question of what exactly is a 'good' reservoir for a given application (Hausler et al., 2003; Ozturk et al., 2007), but no coherent theory has yet emerged. The largely black box character of reservoirs prevents us from performing a deeper theoretical investigation of the dynamical properties of successful reservoirs. Reservoir construction is often driven by a series of (more-or-less) randomised model building stages, with both the researchers and practitioners having to rely on a series of trials and errors. Sometimes reservoirs have been evolved in a costly and difficult to analyse evolutionary computation setting (Bush and Anderson, 2005; Ishii et al., 2004; Schmidhuber et al., 2007; Ajdari Rad et al., 2008).

In chapter 3 we argued that randomisation in reservoir construction may not be necessary. Besides eliminating the problem of non-transparency and trail-and-error construction of standard randomised ESN, the simple deterministically constructed SCR topologies were shown to yield comparable results to ESN on a variety of temporal tasks.

On a number of widely used time series benchmarks of different origin and characteristics, as well as by conducting a theoretical analysis we have shown in chapter 3 that:

1. A very simple cycle topology of reservoir is often sufficient for obtaining performances comparable to those of ESN. Except for the *NARMA* datasets, nonlinear reservoirs were needed.

2. Competitive reservoirs can be constructed in a completely deterministic manner: The reservoir connections all have the same weight value. The input connections have the same absolute value with sign distribution following one of the universal deterministic aperiodic patterns.

3. The memory capacity of linear cyclic reservoirs with a single reservoir weight value r can be made to differ arbitrarily close to the proved optimal value of N, where N

is the reservoir size. In particular, given an arbitrarily small $\epsilon \in (0, 1)$, for

$$r = (1 - \epsilon)^{\frac{1}{2N}},$$

the memory capacity of the cyclic reservoir is $N - \epsilon$.

The simple deterministic nature of our SCR model enabled us to calculate analytically its memory capacity; obtaining such a result for standard ESN is not possible, since:

- for standard ESN one could only calculate the mean memory capacity (with respect to randomisation of ESN construction)
- closed form equality is very difficult to obtain for reservoirs with a range of possible recurrent/input weight values.

Compared with traditional ESN, recent extensions and reformulations of reservoir models often achieved improved performances (Steil, 2007; Xue et al., 2007; Deng and Zhang, 2007), at the price of even less transparent models and less interpretable dynamical organisation. We stress that the main purpose of the work in chapter 3 is not a construction of yet another reservoir model achieving an (incremental or more substantial) improvement over the competitors on the benchmark data sets. Instead, we would like to propose as simplified as possible reservoir construction, without any stochastic component, that while competitive with *standard* ESN, yields transparent models, more amenable to theoretical analysis than the reservoir models proposed in the literature so far.

Such reservoir models can potentially help us to answer the question just what is it in the organisation of the non-autonomous reservoir dynamics that leads to often impressive performances of reservoir computation. Our simple deterministic SCR model introduced in chapter 3 can be used as a a useful baseline in future reservoir computation studies. It is the level of improvement over the SCR baseline that has a potential to truly unveil

the performance gains achieved by the more (and sometimes much more) complex model constructions.

However, in chapter 4 we extended our work in several aspects:

1. We introduced a novel simple deterministic reservoir model, Cycle Reservoir with Jumps (CRJ) with highly constrained weight values, that has superior performance to standard ESN on four temporal tasks of different origin and characteristics.

2. We studied the effect of eigenvalue distribution of the reservoir matrix on the model performance. It has been suggested that a uniform coverage of the unit disk by such eigenvalues can lead to superior model performances. We showed that this is not necessarily so. Despite having highly constrained eigenvalue distribution the CRJ consistently outperformed ESN with much more uniform eigenvalue coverage of the unit disk.

3. We presented a new framework for determining short term memory capacity MC of linear reservoir models to a high degree of precision. Using the framework we showed the effect of shortcut connections in the CRJ reservoir topology on its memory capacity. Due to cross-talk effects introduced by the jumps in CRJ, the MC contributions start to rapidly decrease earlier than in the case of SCR, but unlike in SCR, the decrease in MC_k in CRJ is gradual, enabling the reservoir to keep more information about some of the later inputs.

4. Through the study of pseudo-Lyapunov exponents we showed that even though (unlike ESN) the simple CRJ reservoirs have (average) contractive dynamics, they achieved consistently the best performance. This poses a interesting open question as to whether and in what contexts the "edge-of-chaos" hypothesis can be applied to reservoir computations.

We believe that if given a choice whether to construct a model in a randomised or completely deterministic manner, having guarantees of 'similar' performance levels, it is more advisable to go for the latter. Besides the advantages mentioned above, in our framework the important elements of the model structure have a chance to emerge.
For example, we show that even though simple unidirectional cycle with fixed weight (SCR model) is already competitive, adding regular bidirectional shortcuts (of the same weight) originating and ending in few higher-clustering coefficient nodes (CRJ model), brings potentially huge performance improvements (and sometimes significantly beats ESN). Such an insight could not be obtained using traditional randomised reservoir generation. This opens new research questions as to exactly why such a jump modification has this effect. Such focused research program would not originate from studies consistently using randomised reservoir constructions. On the other hand, using randomised reservoir construction can have beneficial effects on model evaluation - in contrast to deterministically constructed reservoirs, one may need a smaller pool of different tasks to get the same statistical significance.

We propose that in order to quantify the benefit of the potentially complex current or future reservoir formulations, such models should be compared with our simple, deterministically constructed CRJ model that, as shown in chapter 4, has a potential to significantly outperform the traditional ESN.

In chapter 5, we have empirically demonstrated that coupling ESN models through negatively correlated non-linear readouts can lead to performance improvements over the simple ESN ensemble. In contrast to traditional negatively correlated ensembles, the readouts receive different inputs. However, when considering our model as ensemble of ESNs, each receiving the same input stream, the reservoir activations represent internal feature representations of the inputs and the model can be viewed as a novel generalisation of NCL to state space models. There have been studies of simple ESN ensembles (Schwenker and Labib, 2009), or Multi-Layer Perceptron (MLP) readouts (Babinec and

Pospichal, 2006; Bush and Anderson, 2005), but to the best of our knowledge, this is the first study employing a NCL style training in ensembles of state space models, such as ESNs.

Finally, in Chapter 6 , we have shown that under some assumptions, the two quantities measures suggested to characterise short term memory in input driven dynamical systems, namely the short term memory capacity spectrum MC_k and the Fisher memory curve $J(k)$ can be interpreted as squared 'Mahalanobis' norms of images of the input vector under the system's dynamics, and that even though MC and FMC map the memory structure of the system from two quite different perspectives, they can be linked by a close relation.

7.2 Future Work

Several future work directions arise to extend the results of this work. Here we will introduce some ideas we are planning to explore in our future research work.

7.2.1 Reservoir characterisations

It seems that characterisations of reservoirs in terms of memory capacity, eigenvalue decomposition of the reservoir weight matrix or pseudo-Lyapunov exponents, cannot easily capture what makes reservoirs great temporal modelling tools. Reservoirs are non-linear non-autonomous dynamical systems that are difficult to characterise by linearisation techniques (eigenspectrum), or methods not directly representing task-related useful temporal structure in the input driving stream (memory capacity). Theory and practice of deep reservoir characterisations that can be directly linked to their performance is an open problem for future work.

Moreover, in contrast to the complex trial-and-error ESN construction, our simple

approach (Simple Cycle Reservoir (SCR)) introduced in Chapter 3 leaves the user with only two free parameters to be set, r and v. This not only considerably simplifies the ESN construction, but also enables a more thorough theoretical analysis of the reservoir characterisations. The doors can be open for a wider acceptance of the ESN methodology amongst both practitioners and theoreticians working in the field of time series modelling/prediction. In addition, our simple deterministically constructed reservoir models (SCR and CRJ) can serve as useful baselines in future reservoir computing studies. The simple nature of our SCR reservoir can enable a systematic study of the short-term Memory Capacity (MC) measure for different kinds of input stream sources and this is a matter for future work.

7.2.2 Input weight and reservoir structures

Even though the theoretical analysis of the Simple Cycle Reservoir (SCR) introduced in Chapter 3 has been done for the linear reservoir case, the requirement that all cyclic rotations of the input vector need to be linearly independent seem to apply to the non-linear case as well. Indeed, under the restriction that all input connections have the same absolute weight value, the linear independence condition translates to the requirement that the input sign vector follows an aperiodic pattern. Of course, from this point of view, a simple standard basis pattern (+1,-1,-1,...,-1) is sufficient. Interestingly enough, we found out that the best performance levels were obtained when the input sign pattern contained roughly equal number of positive and negative signs. At the moment we have no satisfactory explanation for this phenomenon and we leave it as an open question for future work.

Moreover, for the moment in the case of bi-directional regular jumps (CRJ), we don't have an explanation for why we need to start from the same landing jump n not from the next unit $n + 1$ for the landing jump to achieve better results, and we leave this as an open question for future work.

7.2.3 Negative Correlation Learning through time

Negative Correlation Learning (NCL) is a successful ensemble technique by inducing diversity among ensemble members explicitly. This has been verified in several studies on static data (no dependencies of inputs through time). In chapter 5, we designed a Negatively Correlated Ensemble of Echo state Networks, where Negative correlation learning achieved useful results, and as a future work, it is good to extend the idea to input dependencies through time, so we can train an Ensemble of Recurrent Neural Networks (RNNs) using BPTT or RTRL , where all the individual RNNs are trained simultaneously and interactively through the correlation penalty terms in their error functions.

7.3 Chapter Summary

We have drawn the conclusions in section 7.1, where these conclusions are discussed in details. Several research directions were given in section 7.2.

Appendix A

Experimental Setup and Detailed Results

General description of the experimental setup used in section 3.2 is summarised in table A.1, with details on selected model parameters for different data sets presented in table A.2. Detailed results including standard deviations across repeated experiments (as described in chapter 3 section 3.2) are shown in tables A.3 : A.16.

Table A.1: Experimental Setup

Datasets	NARMA (of different orders), Santa Fe Laser, Hénon Map, Nonlinear Communication Channel, Sunspots, IPIX Radar, Nonlinear System with Observational Noise, and Isolated Digits
Model class topologies	ESN, DLR, DLRB, and SCR
Readout learning	RLS with dynamic noise injection , and Ridge Regression
Reservoir weights	ESN: (random weights with spectral radius $\alpha = [0.05 : 0.05 : 1]$, and connectivity $con = [0.05 : 0.05 : 0.5]$) DLR, DLRB , and SCR: ($r = [0.05 : 0.05 : 1]$, $b = [0.05 : 0.05 : 1]$) where $b \in 1 - r < b < 1/(4r)$
reservoir sizes	$[50 : 50 : 200]$ In case of *IPIX Radar* and sunspots $N = 80$ and $N = 200$, respectively.
input scale	$[0.01 : 0.005 : 1]$
input sign generation	(1) random draw from Bernoulli distribution (mean=1/2), (2) decimal expansion of irrational numbers (π and e), (3) binary symbolic dynamics of the logistic map
noise size for RLS	$[0 : 10^{-0.25} : 10^{-15}]$
generalisation factor for Ridge regression	$[0 : 10^{-0.25} : 10^{-15}]$

Table A.2: Selected Model Parameters Based on the Validation Set Performance

Dataset	Item	ESN	DLR	DLRB	SCR
NARMA $N = 100$	Input weight connection	uniform over (-0.1,0.1)	±0.1	±0.1	±0.1
	reservoir weights	$\alpha = 0.95$	r=0.8	r=0.8, b=0.05	r=0.8
	Sparseness of W	0.1	-	-	-
Laser $N = 100$	Input weight connection	uniform over (-1,1)	±0.6	±0.6	±0.6
	reservoir weights	$\alpha = 0.95$	r=1	r=1, b=0.01	r=1
	Sparseness of W	0.5	-	-	-
Hénon Map $N = 100$	Input weight connection	uniform over (-1,1)	±0.95	±0.95	±0.95
	reservoir weights	$\alpha = 0.3$	r=0.95	r=0.95, b=0.05	r=0.95
	Sparseness of W	0.5	-	-	-
Nonlinear Communication Channel $N = 100$	Input weight connection	uniform over (-0.025,0.025)	±0.025	±0.025	±0.025
	reservoir weights	$\alpha = 0.5$	r=0.95	r=0.95, b=0.05	r=0.95
	Sparseness of W	0.2	-	-	-
Sunspots $N = 200$	Input weight connection	uniform over (-1,1)	±1	±1	±1
	reservoir weights	$\alpha = 0.75$	r=0.3	r=0.3, b=0.1	r=0.3
	Sparseness of W	0.2	-	-	-
Nonlinear System with Observational Noisy $N = 100$	Input weight connection	uniform over (-0.1,0.1)	±0.025	±0.025	±0.025
	reservoir weights	$\alpha = 0.65$	r=0.65	r=0.65, b=0.2	r=0.65
	Sparseness of W	0.2	-	-	-
IPIX Radar $N = 80$	Input weight connection	uniform over (-0.04,0.04)	±0.04	±0.04	±0.04
	reservoir weights	$\alpha = 0.7$	r=0.65	r=0.6, b=0.05	r=0.65
	Sparseness of W	0.13	-	-	-
Isolated Digits $N = 100$	Input weight connection	uniform over (-1,1)	±1	±1	±1
	reservoir weights	$\alpha = 1$	r=0.1	r=0.1, b=0.05	r=0.1
	Sparseness of W	0.8	-	-	-

Table A.3: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *10th order NARMA* dataset for internal nodes with *tanh* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.166 (0.0171)	0.163 (0.0138)	0.158 (0.0152)	0.160 (0.0134)
100	0.0956 (0.0159)	0.112(0.0116)	0.105 (0.0131)	0.0983 (0.0156)
150	0.0514 (0.00818)	0.0618 (0.00771)	0.0609 (0.00787)	0.0544 (0.00793)
200	0.0425 (0.0166)	0.0476 (0.0104)	0.0402 (0.0110)	0.0411 (0.0148)

Table A.4: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *10th order NARMA* dataset for internal nodes with *linear* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.1601 (6.108E-04)	0.1606 (8.342E-05)	0.1602 (3.889E-04)	0.1603 (1.196E-04)
100	0.1602 (4.152E-04)	0.1607 (6.574E-05)	0.1600 (2.916E-04)	0.1603 (6.940E-05)
150	0.1603 (3.401E-04)	0.1607 (3.760E-05)	0.1599 (2.715E-04)	0.1603 (2.167E-05)
200	0.1604 (3.612E-04)	0.1606 (6.437E-05)	0.1599 (3.930E-04)	0.1603 (2.610E-05)

Table A.5: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *10th order random NARMA* dataset for internal nodes with *tanh* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.131 (0.0165)	0.133 (0.0132)	0.130 (0.00743)	0.129 (0.0111)
100	0.0645 (0.0107)	0.0822 (0.00536)	0.0837 (0.00881)	0.0719 (0.00501)
150	0.0260 (0.0105)	0.0423 (0.00872)	0.0432 (0.00933)	0.0286 (0.00752)
200	0.0128 (0.00518)	0.0203 (0.00536)	0.0201 (0.00334)	0.0164 (0.00412)

Table A.6: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *10th order random NARMA* dataset for internal nodes with *linear* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.1497 (3.033E-04)	0.1502 (3.916E-04)	0.1501 (2.178E-04)	0.1501 (2.574E-04)
100	0.1499 (2.219E-04)	0.1500 (2.232E-04)	0.1496 (1.912E-04)	0.1501 (2.557E-04)
150	0.1499 (2.782E-04)	0.1502 (3.264E-04)	0.1498 (2.170E-04)	0.1501 (3.706E-04)
200	0.1500 (3.217E-04)	0.1502 (1.753E-04)	0.1497 (1.820E-04)	0.1501 (1.466E-04)

Table A.7: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *20th order NARMA* dataset for internal nodes with *tanh* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.297 (0.0563)	0.232 (0.0577)	0.238 (0.0507)	0.221 (0.0456)
100	0.235 (0.0416)	0.184 (0.0283)	0.183 (0.0196)	0.174 (0.0407)
150	0.178 (0.0169)	0.171 (0.0152)	0.175 (0.0137)	0.163 (0.0127)
200	0.167 (0.0164)	0.165 (0.0158)	0.160 (0.0153)	0.158 (0.0121)

Table A.8: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *20th order NARMA* dataset for internal nodes with *linear* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.1446 (9.922E-04)	0.1441 (1.624E-04)	0.1428 (3.668E-04)	0.1439 (8.446E-04)
100	0.1437 (3.866E-04)	0.1430 (1.133E-04)	0.1426 (4.284E-05)	0.1431(7.762E-05)
150	0.1434 (4.601E-04)	0.1430 (5.243E-05)	0.1426 (4.636E-05)	0.1430 (3.017E-05)
200	0.1433 (3.787E-04)	0.1430 (4.148E-05)	0.1426 (5.896E-05)	0.1430 (3.620E-05)

Table A.9: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *laser* dataset for internal nodes with *tanh* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.0184 (0.00231)	0.0210 (0.00229)	0.0215 (0.00428)	0.0196 (0.00219)
100	0.0125 (0.00117)	0.0132 (0.00116)	0.0139 (0.00121)	0.0131 (0.00105)
150	0.00945 (0.00101)	0.0107 (0.00114)	0.0112 (0.00100)	0.0101 (0.00109)
200	0.00819 (5.237E-04)	0.00921 (9.122E-04)	0.00913 (9.367E-04)	0.00902 (6.153E-04))

Table A.10: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *Hénon Map* dataset for internal nodes with *tanh* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.00975 (0.000110)	0.0116 (0.000214)	0.0110 (0.000341)	0.0106 (0.000185)
100	0.00894 (0.000122)	0.00982 (0.000143)	0.00951 (0.000120)	0.00960 (0.000124)
150	0.00871 (4.988E-05)	0.00929 (6.260E-05)	0.00893 (6.191E-05)	0.00921 (5.101E-05)
200	0.00868 (8.704E-05)	0.00908 (9.115E-05)	0.00881 (9.151E-05)	0.00904 (9.250E-05)

Table A.11: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *Nonlinear Communication Channel* dataset for internal nodes with *tanh* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.0038 (4.06E-4)	0.0034 (2.27E-4)	0.0036 (2.26E-4)	0.0035 (2.55E-4)
100	0.0021 (4.42E-4)	0.0015 (1.09E-4)	0.0016 (1.07E-4)	0.0015 (1.23E-4)
150	0.0015 (4.01E-4)	0.0011 (1.12E-4)	0.0011 (1.08E-4)	0.0012 (1.23E-4)
200	0.0013 (1.71E-4)	0.00099 (6.42E-5)	0.0010 (7.41E-5)	0.0010 (7.28E-5)

Table A.12: Test set performance of ESN, SCR, DLR, and DLRB topologies on the *Isolated Digits* dataset for internal nodes with *tanh* transfer function f.

reservoir Size	ESN	DLR	DLRB	SCR
50	0.0732 (0.0193)	0.0928 (0.0177)	0.1021 (0.0204)	0.0937 (0.0175)
100	0.0296 (0.0063)	0.0318 (0.0037)	0.0338 (0.0085)	0.0327 (0.0058)
150	0.0182 (0.0062)	0.0216 (0.0052)	0.0236 (0.0050)	0.0192 (0.0037)
200	0.0138 (0.0042)	0.0124 (0.0042)	0.0152 (0.0038)	0.0148 (0.0050)

Table A.13: Test set performance of SCR topology on the *20th order NARMA* dataset using three different ways of generating pseudo-randomised input sign patterns: initial digits of π and *Exp*; symbolic dynamics of logistic map.

reservoir Size	ESN	SCR-PI	SCR-Ex	SCR-Log
50	0.297 (0.0563)	0.233 (0.0153)	0.232 (0.0175)	0.196 (0.0138)
100	0.235 (0.0416)	0.186 (0.0166)	0.175 (0.0136)	0.169 (0.0172)
150	0.178 (0.0169)	0.175 (0.00855)	0.158 (0.0103)	0.156 (0.00892)
200	0.167 (0.0164)	0.166 (0.00792)	0.157 (0.00695)	0.155 (0.00837)

Table A.14: Test set performance of SCR topology on the *laser* dataset using three different ways of generating pseudo-randomised input sign patterns: initial digits of π and *Exp*; symbolic dynamics of logistic map.

reservoir Size	ESN	SCR-PI	SCR-Ex	SCR-Log
50	0.0184 (0.00231)	0.0204	0.0187	0.0181
100	0.0125 (0.00117)	0.0137	0.0153	0.0140
150	0.00945 (0.00101)	0.0115	0.0111	0.0126
200	0.00819 (5.237E-04)	0.00962	0.00988	0.0107

Table A.15: Test set performance of SCR topology on the *Hénon Map* dataset using three different ways of generating pseudo-randomised input sign patterns: initial digits of π and *Exp*; symbolic dynamics of logistic map.

reservoir Size	ESN	SCR-PI	SCR-Ex	SCR-Log
50	0.00975 (0.000110)	0.00986	0.00992	0.00998
100	0.00894 (0.000122)	0.00956	0.00985	0.00961
150	0.00871 (4.988E-05)	0.00917	0.00915	0.00920
200	0.00868 (8.704E-05)	0.00892	0.00883	0.00898

Table A.16: Test set performance of SCR topology on the *Non-linear Communication Channel* dataset using three different ways of generating pseudo-randomised input sign patterns: initial digits of π and *Exp*; symbolic dynamics of logistic map.

reservoir Size	ESN	SCR-PI	SCR-Ex	SCR-Log
50	0.0038 (4.06E-4)	0.0036 (1.82E-04)	0.0026 (6.23E-05)	0.0033 (1.09E-04)
100	0.0021 (4.42E-4)	0.0016 (7.96E-05)	0.0017 (1.04E-04)	0.0015 (8.85E-5)
150	0.0015 (4.01E-4)	0.0012 (7.12E-05)	0.0011 (6.10E-05)	0.0012 (4.56E-05)
200	0.0013 (1.71E-4)	0.00088 (2.55E-05)	0.00090 (3.05E-05)	0.00093 (3.33E-05)

Appendix B

Selected model representatives

In this appendix we show detailed parameter settings of the selected model representatives for our experiments in chapter 4. Details of parameter values of models used in section 4.2 are provided in Table B.1. Table B.2 reports parameters for models used in comparison experiment with SWNR (section 4.3). Finally, we report parameter values of the selected hierarchical extension (CRHJ) of the CRJ model in Table B.3 (section 4.3).

Table B.1: Parameter Values for the Selected ESN, SCR and CRJ Model Representatives with Reservoir Sizes of N

Dataset	ESN	SCR	CRJ
laser $N = 200$	$con = 0.2$, $\lambda = 0.95$, $a = 1$	$v = 0.85$, $r_c = 0.7$	$v = 0.9$, $r_c = 0.7$, $r_j = 0.4$, $\ell = 5$
NARMA $N = 200$	$con = 0.15$, $\lambda = 0.85$, $a = 0.1$	$v = 0.05$, $r_c = 0.8$	$v = 0.05$, $r_c = 0.7$, $r_j = 0.5$, $\ell = 5$
speech $N = 200$	$con = 0.4$, $\lambda = 0.95$, $a = 1$	$v = 1$, $r_c = 0.95$	$v = 1$, $r_c = 0.9$, $r_j = 0.4$, $\ell = 13$
memory and nonlinear mapping task $N = 100$	$con = 0.2$, $\lambda = 0.95$, $a = 0.05$	$v = 0.025$, $r_c = 0.7$	$v = 0.025$, $r_c = 0.8$, $r_j = 0.3$, $\ell = 24$

Table B.2: Parameter Values for the Selected ESN, SWNR, SCR and CRJ Model Representatives (Reservoir Size $N = 500$).

Dataset	ESN	SWNR	SCR	CRJ
laser	$con = 0.15$, $\lambda = 0.9$, $a = 1$	$\lambda = 5.5$, $a = 1$	$v = 0.7$, $r_c = 0.75$	$v = 0.7$, $r_c = 0.75$, $r_j = 0.15$, $\ell = 10$
NARMA	$con = 0.2$, $\lambda = 0.95$, $a = 0.1$	$\lambda = 2$, $a = 0.2$	$v = 0.05$, $r_c = 0.8$	$v = 0.1$, $r_c = 0.8$, $r_j = 0.5$, $\ell = 21$

Table B.3: Parameter Values for the Selected CRHJ Model Representative (Reservoir Size $N = 100$).

Dataset	CRHJ
NARMA	$v = 0.05$, $r_c = 0.6$, $r_{j_1} = 0.05$, $r_{j_2} = 0.4$, $r_{j_3} = 0.25$
laser	$v = 1$, $r_c = 1$, $r_{j_1} = 0.55$, $r_{j_2} = 0.4$, $r_{j_3} = 0.1$

Bibliography

A. Ajdari Rad, M. Jalili, and M. Hasler. Reservoir optimization in recurrent neural networks using kronecker kernels. In *IEEE ISCAS*, 2008.

A. F. Atiya and A. G. Parlos. New results on recurrent network training: Unifying the algorithms and accelerating convergence. *IEEE Transactions on Neural Networks*, 11: 697–709, 2000.

S. Babinec and J. Pospichal. Merging echo state and feedforward neural networks for time series forecasting. In *Proceedings of the 16th International Conference on Artificial Neural Networks (ICANN 2006), volume 4131 of LNCS, pages 367-375. Springer*, 2006.

A. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286: 509–512, 1999.

S. Basterrech, C. Fyfe, and G. Rubino. Self-organizing maps and scale-invariant maps in echo state networks. In *11th International Conference on Intelligent Systems Design and Applications (ISDA)*, 2011.

Y. Bengio and Y. LeCun. Scaling learning algorithms toward AI. *In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, Large Scale Kernel Machines. MIT Press, Cambridge, MA*, 2007.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

N. Bertschinger and T. Natschlager. Real-time computation at the edge of chaos in recurrent neural networks. *Neural Computation*, 16(7):1413–1436, 2004.

G. Brown. *Diversity in Neural Network Ensembles.* PhD thesis, School of Computer Science, University of Birmingham, 2004.

G. Brown and X. Yao. On the effectiveness of negative correlation learning. In *First UK Workshop on Computational Intelligence (UKCI'01), Edinburgh, Scotland*, 2001.

G. Brown, J. L. Wyatt, , and P. Tino. Managing diversity in regression ensembles. *Journal of Machine Learning Research*, 6:1621–1650, 2005.

K. Bush and C. Anderson. Modeling reward functions for incomplete state representations via echo state networks. In *Proceedings of the International Joint Conference on Neural Networks, Montreal, Quebec*, 2005.

M. Cernansky and M. Makula. Feed-forward echo state networks. In *Proceedings of the IEEE International Joint Conference on Neural Networks, 2005 (IJCNN 2005),1479-1482*, 2005.

M. Cernansky and P. Tino. Predictive modelling with echo state networks. In *Proceedings of the 18th international conference on Artificial Neural Networks, (eds) V. Kurkova, R. Neruda, J. Koutnik. pp. 778-787, Lecture Notes in Computer Science, LNCS 5163, Springer-Verlag*, 2008.

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

K.P. Dockendorf, I.I. Park, H. Ping, J.C. Principe, and T.B. DeMarse. Liquid state machines and cultured cortical networks: The separation property. *Biosystems*, 95(2): 90–97, 2009.

X. Dutoit, B. Schrauwen, J. Van Campenhout, D. Stroobandt, H. Van Brussel, and M. Nuttin. Pruning and regularization in reservoir computing. *Neurocomputing*, 72: 1534–1546, 2009.

G. Fette and J. Eggert. Short term memory and pattern matching with simple echo state networks. In *Proc. of ICANN, pp. 1318*, 2005.

S. Ganguli, D. Huh, and H. Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences*, 105:18970–18975, 2008.

F.A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12:2451–2471, 1999.

N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEEE Proc.-Radar, Sonar Navig., vol. 140, pp. 107-113*, 1993.

U. Gunturkun and H. Bruin. A comparative study on the iterative prediction with ESN and RBF network. *IEEE SIGNAL PROCESSING LETTERS*, 2008.

B. Hammer and P. Tino. Recurrent neural networks with small weights implement definite memory machines. *Neural Computation*, 15(8):1897–1926, 2003.

B. Hammer, B. Schrauwen, and J.J. Steil. Recent advances in efficient learning of recurrent networks. In *17th European Symposium on Artificial Neural Networks (ESANN 2009), Bruges, Belgium*, 2009.

S. Hausler, M. Markram, and W. Maass. Perspectives of the high-dimensional dynamics of neural microcircuits from the point of view of low-dimensional readouts. *Complexity (Special Issue on Complex Adaptive Systems)*, 8(4):39–50, 2003.

S. Haykin. *Adaptive filter theory (4th ed.)*. Englewood Cliffs, NJ:prentice-Hall., 2001.

S. Haykin. *Neural Networks: A Comprehensive Foundation (2nd ed.)*. Upper Saddle River, NJ: Prentice Hall, 2nd Edition, 1999.

M. Henon. A two-dimensional mapping with a strange attractor. *Comm. Math. Phys.*, 50:69–77, 1976.

G. Holzmann and H. Hauser. Echo state networks with filter neurons and a delay and sum readout. *Neural Networks*, 32(2):244–256, 2009.

K. Ishii, T. van der Zant, V. Becanovic, and P. Ploger. Identification of motion with echo state network. In *Proceedings of the OCEANS 2004 MTS/IEEE -TECHNO-OCEAN Conference, volume 3, pages 1205-1210*, 2004.

H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. Technical report GMD report 148, German National Research Center for Information Technology, 2001.

H. Jaeger. Short term memory in echo state networks. Technical report GMD report 152, German National Research Center for Information Technology, 2002a.

H. Jaeger. Adaptive nonlinear systems identification with echo state network. *Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA.*, 15:593–600, 2003.

H. Jaeger. Reservoir riddles: Suggestions for echo state network research. In *Proceedings of International Joint Conference on Neural Networks IJCNN 2005, Montreal, Canada. (1460-1462)*, 2005.

H. Jaeger. A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach. Technical report GMD report 159, German National Research Center for Information Technology, 2002b.

H. Jaeger. Discovering multiscale dynamical features with hierarchical echo state networks. Technical report, Jacobs University technical report Nr. 10, 2007.

H. Jaeger and H. Hass. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. *Science*, 304:78–80, 2004.

H. Jaeger, M. Lukosevicius, D. Popovici, and U. Siewert. Optimisation and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352, 2007a.

H. Jaeger, W. Maass, and J.C. Principe. Special issue. *Neural Networks*, 20, 2007b.

B. Jones, D. Stekel, J. Rowe, and C. Fernando. Is there a liquid state machine in the bacterium escherichia coli? In *Proceedings of the 2007 IEEE Symposium on Artificial Life (CI-Alife), pages 187-191.*, 2007.

M. Kaiser and C.C. Hilgetag. Spatial growth of real-world networks,. *Phys. Rev.*, E69: 036103, 2004.

R. Legenstein and W Maass. Edge of chaos and prediction of computational performance for neural circuit models. *Neural Networks*, 20(3):323–334, 2007.

R. Legenstein and W. Maass. What makes a dynamical system computationally powerful? *New Directions in Statistical Signal Processing: From Systems to Brain, Cambridge, MA:MIT Press*, 2005.

B. Lieblad. Exploration of effects of different network topologies on the esn signal crosscorrelation matrix spectrum. Master's thesis, Bachelor's thesis, Jacobs University Bremen, 2004.

Y. Liu and X. Yao. Ensemble learning via negative correlation. *Neural Networks*, 12: 1399–1404, 1999.

Y. Liu, X. Yao, and T. Higuchi. Evolutionary ensembles with negative correlation learning. *IEEE Transactions on Evolutionary Computation*, 4:380–387, 2000.

Y. Lui. *Negative Correlation learning and evolutionary Neural Network Ensembles*. PhD thesis, The University of New South wales, Australian Defence Force Academy, 1998.

M. Lukosevicius. Echo state networks with trained feedbacks. technical report no. 4. Technical report, Jacobs University Bremen, 2007.

M. Lukosevicius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

M. Lukosevicius and H. Jaeger. Overview of reservoir recipes. Technical report no. 11, School of Engineering and Science, Jacobs University, 2007.

R.F. Lyon. A computational model of filtering, detection and compression in the cochlea. In *Proceedings of the IEEE ICASSP, pages 1282-1285*, 1982.

W. Maass, T. Natschlager, and H. Markram. Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

W. Mass, T. Natschlager, and H. Markram. Fading memory and kernel properties of generic cortical microcircuit models. *Journal of Physiology*, 98(4-6):315–330, 2004.

N. Mayer and M. Browne. Echo state networks and self-prediction. In *First International Workshop, Biologically Inspired Approaches to Advanced Information Technology, BioADIT, pp. 40-48 Lecture Notes in Computer Science, LNCS 3141, Springer Berlin/ Heidelberg*, 2004.

R. Mckay and H. Abbass. Analysing anticorrelation in ensemble learning. In *Proceedings of 2001 conference on Artificial Neural Networks and Expert systems, pp. 22-27*, 2001.

M. E. J. Newman and D. J. Watts. Scaling and percolation in the small-world network model. *Phys. Rev.*, 60:7332–7342, 1999.

M. Ozturk and J. Principe. An associative memory readout for esns with application to dynamical pattern recognition. *Neural Network*, 20:377–390, 2007.

M. C. Ozturk, D. Xu, and J. Principe. Analysis and design of echo state network. *Neural Computation*, 19(1):111–138, 2007.

D. Prokhorov. Echo state networks: appeal and challenges. In *Proc. of International Joint Conference on Neural Networks (pp. 1463-1466). Montreal, Canada.*, 2005.

D. Prokhorov, G. Puskorius, and Feldkamp L. Dynamical neural networks for control. In *A Field Guide to Dynamic Recurrent Networks, IEEE Press*, 2001.

A. Rodan and A. Tino. Negatively correlated echo state networks. In *proceedings of the 19th European Symposium on Artificial Neural Networks - Computational Intelligence and Machine Learning ESANN 2011. Bruges (Belgium), 27-29 April 2011, d-side publi*, 2011b.

A. Rodan and P. Tino. Minimum complexity echo state network. *IEEE Transactions on Neural Networks*, 22(1):131–144, 2011.

A. Rodan and P. Tino. Simple deterministically constructed recurrent neural networks. In *Intelligent Data Engineering and Automated Learning (IDEAL 2010), pp. 267-274, Lecture Notes in Computer Science, LNCS 6283, Springer-Verlag*, 2010.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. *Parallel distributed processing: explorations in the microstructure of cognition*, 1: foundations,:318–362, 1986.

U. Schiller and J. Steil. Analysing the weight dynamics of recurrent learning algorithms. *Neurocomputing*, 63C:5–23, 2005.

J. Schmidhuber, D. Wierstra, M. Gagliolo, and F. Gomez. Training recurrent networks by evolino. *Neural Computation*, 19:757–779, 2007.

B. Schrauwen and J.V. Campenhout. Linking non-binned spike train kernels to several existing spike train metrics. In *M. Verleysen, editor, Proceedings of the 14th European Symposium on Artificial Neural Networks (ESANN2006),pages 41-46, Evere*, 2006.

B. Schrauwen, J. Defour, D. Verstraeten, and J.M. Van Campenhout. The introduction of time-scales in reservoir computing, applied to isolated digits recognition. In *Proceedings of the 17th International Conference on Artificial Neural Networks (ICANN 2007), volume 4668 of LNCS, pages 471-479. Springer*, 2007a.

B. Schrauwen, D. Verstraeten, and J.V. Campenhout. An overview of reservoir computing: theory, applications and implementations. In *ESANN'2007 proceedings - European Symposium on Artificial Neural Networks, Bruges, Belgium, 471-482*, 2007b.

B. Schrauwen, L. Buesing, and R. A. Legenstein. On computational power and the order-chaos phase transition in reservoir computing. In *Neural Information Processing Systems (NIPS), 425-1432*, 2008a.

B. Schrauwen, M. Wardermann, D. Verstraeten, J.J. Steil, and D Stroobandt. Improving reservoirs using intrinsic plasticity. *Neurocomputing*, 71(7-9):1159–1171, 2008b.

F. Schwenker and A. Labib. Echo state networks and neural network ensembles to predict sunspots activity. In *ESANN 2009 proceedings, European Symposium on Artificial Neural Networks -Advances in Computational Intelligence and Learning, Bruges (Belgium)*, 2009.

S. Singhal and L. Wu. Training multilayer perceptrons with the extended kalman algorithm. *Advances in Neural Information Processing Systems*, 1:133–140, 1989.

M.D. Skowronski and J.G. Harris. Minimum mean squared error time series classification using an echo state network prediction model. In *IEEE International Symposium on Circuits and Systems, Island of Kos, Greece, pp. 3153-3156*, 2006.

M. Slutzky, P. Cvitanovic, and D. Mogul. Manipulating epileptiform bursting in the rat hippocampus using chaos control and adaptive techniques. *IEEE transactions on bio-medical engineering*, 50(5):559–570, 2003.

J. Steil. Online stability of backpropagation-decorrelation recurrent learning. *Neurocomputing*, 69:642–650, 2006.

J. Steil. Online reservoir adaptation by intrinsic plasticity for backpropagation-decorrelation and echo state learning. *Neural Networks*, 20:353–364, 2007.

J. J. Steil. Backpropagation-decorrelation: Recurrent learning with O(N) complexity. In *Proc. IJCNNNeural Networks, 2004. IJCNN '04. Proceedings. 2004 IEEE International Joint Conference on, volume 2, pp. 843-848*, 2004.

W. Tabor. The value of symbolic computation. *Ecological Psychology*, 14(1-2):21–51, 2002.

P. Tino and G. Dorffner. Predicting the future of discrete sequences from fractal representations of the past. *Machine Learning*, 45(2):187–218, 2001.

M. H. Tong, A.D. Bicket, E.M. Christiansen, and G.W. Cottrell. Learning grammatical structure with echo state network. *Neural Networks*, 20:424–432, 2007.

F. Triefenbach, A. Jalalvand, B. Schrauwen, and J. Martens. Phoneme recognition with large hierarchical reservoirs. *In Advances in Neural Information Processing Systems (NIPS 2010)*, 23:9, 2010.

J. Triesch. Synergies between intrinsic and synaptic plasticity in individual model neurons. In *Proc. of NIPS, vol. 17*, 2004.

T. van der Zant, V. Becanovic, K. Ishii, H.U Kobialka, and P.G Pl oger. Finding good echo state networks to control an underwater robot using evolutionary computations. In *Proc. of IAV*, 2004.

D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt. The unified reservoir computing concept and its digital hardware implementations. In *Proc. LATSIS, pages 139-140*, 2006.

D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20:391–403, 2007.

D. Verstraeten, J. Dambre, X. Dutoit, and B. Schrauwen. Memory versus non-linearity in reservoirs. In *2010 International Joint Conference on Neural Networks (IJCNN), pp. 1-8. IEEE Press*, 2010.

T. Wang and C. Fyfe. Training echo state networks with neuroscale. In *2011 International Conference on Technologies and Applications of Artificial Intelligence*, 2011.

D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

P.J Werbos. Backpropagation through time: what it does and how to do it. In *Proceedings of the IEEE, 78(10):1550-1560*, 1990.

R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280, 1989.

F. Wyffels, B. Schrauwen, , and D. Stroobandt. Stable output feedback in reservoir computing using ridge regression. In *Proceedings of the 18th international conference on Artificial Neural Networks, pp.808-817, Lecture Notes in Computer Science, LNCS 5163, Springer-Verlag*, 2008.

Y. Xue, L. Yang, and S. Haykin. Decoupled echo state networks with lateral inhibition. *Neural Networks*, 20:365–376, 2007.

X. Yao, M. Fischer, and G. Brown. Neural network ensembles and their application to traffic flow prediction in telecommunications networks. In *proceedings of international Joint Conference on Neural Networks, pp. 693-698. IEEE Press*, 2001.

B. Zhang and Y. Wang. Echo state networks with decoupled reservoir states. In *18th IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2008)*, 2008.

Shi Zhinwei and Han. Min. Support vector echo-state machine for chaotic time-series prediction. *IEEE Transactions on Neural Networks*, 18(2):359–72, 2007.