EFFECTS OF NETWORK WEIGHT STRUCTURE IN ECHO STATE NETWORKS

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Science and Engineering

2020

Danny J. Wood
School of Engineering
Department of Computer Science

Contents

Word Count: 41,553

List of Tables

List of Figures

List of Notation

Echo State Networks

W	Recurrent weight matrix
$\mathbf{v}$	Input weight vector
V	Input weight matrix
M	Size of hidden state
D	Dimension of elements of the input sequence
$\mathbf{x}_t$	Hidden state of network at time t
$\mathbf{u}_t$	Vector input at time t
u_t	Scalar input at time t
MC_k	The k-delay memory capacity of the network
MC	The total memory capacity of the network
$\mathbf{w}_k$	The optimal reconstruction weights for u_{t-k}
$(W, \mathbf{v})$	The linear Echo state network with recurrent weight matrix W and input weight vector $\mathbf{v}$
$M_{W,\mathbf{v}}$	The memory matrix of the network $(W, \mathbf{v})$
C_{state}	Bound on the norm of the hidden state of the network
C_{in}	Bound on the norm of the inputs
P	Matrix of eigenvectors of W
Λ	Matrix whose diagonal entries are the eigenvalues of W
J	The Jordan normal form of the matrix W

$\mathbf{v}$'	Input weight vector in the basis of (generalised) eigenvectors of W
α	Leaky integrator parameter
$G(\mathbf{x}, U)$	Iterated update operator acting on initial state $\mathbf{x}$ and finite-length sequence U

Deep Echo State Networks

$W_1, \ldots W_L$	The recurrent weight matrices for layers $1, ..., L$ of the network
$\mathbf{v}_1$	The input vector for the first later of the network
$V_2, \ldots V_L$	The input weight matrices for layers $2, ..., L$ of the network
$\mathbf{x}_t^{(l)}$	The state of the lth layer of the network at time t
$\bar{W}$	The recurrent weight matrix of the flattened network
$\bar{\mathbf{v}}$	The input weight matrix of the flattened weight matrix
$\bar{W}^{(l)}$	Recurrent matrix for the flattened network composed of the first l layers of $(\bar{W}, \bar{\mathbf{v}})$
$\bar{\mathbf{v}}^{(l)}$	Input weight vector for the flattened network composed of the first l layers of $(\bar{W}, \bar{\mathbf{v}})$
$\bar{\mathbf{x}}_t^{(l)}$	The hidden state at time t for the flattened network composed of the first l layers of $(\bar{W}, \bar{\mathbf{v}})$
$\bar{\mathbf{y}}_k^{(l)}$	The contribution of a unit input k time-steps ago to the lth layer of the network in the present
L	The number of hidden layers in the network
$MC_k^{(l)}$	The k-delay memory capacity of the lth layer of the network
$MC^{(l)}$	The total memory capacity of the lth layer of the network

Miscellaneous

$\rho(X)$	Spectral radius of the matrix X
$X^\intercal$	Transpose of the matrix X
$d(\mathbf{x}, \mathbf{y})$	Euclidean distance between vectors $\mathbf{x}$ and $\mathbf{y}$

δ_{ij}	The Kronecker delta (i.e., 1 when $i = j$, zero otherwise)
$\|\mathbf{v}\|_P$	The infinity norm of $\mathbf{v}$ in the basis of columns of Q
$r_1, r_2, \ldots$	Independent Rademacher random variables
σ^2	Variance or proxy-variance of a distribution
$o(f(n))$	A function, say $g(n)$, such that $lim_{n\to\infty} g(n)/f(n)$
$\det(A)$	Determinant of the matrix A
$\mathrm{var}(X)$	Variance of the random variable X
$\mathrm{cov}(X)$	Covariance of the random variable X
S^{M-1}	The set of all unit-norm vectors in the space $\mathbb{R}^M$
$\mathrm{diag}(A_1, A_2, ...)$	Matrix whose diagonal consists of the matrix $A_1, A_2, \ldots$ in order, and whose other entries are all zero
$S(A)$	The set of eigenvalues of the matrix A

List of Abbreviations

ALR	Adjacent-feedback Loop Reservoir
ANN	Artificial Neural Network
cov	Covariance
DeepESN	Deep Echo State Network
DFT	Discrete Fourier Transform
DLR	Delay Line Reservoir
ESN	Echo State Network
ESP	Echo State Property
FNN	Feedforward Neural Network
GRU	Gated Recurrent Unit
i.i.d.	independent and identically distributed
Lin-DeepESN	Linear Deep Echo State Network
LSTM	Long Short Term Memory
MC	Memory Capacity
MC_k	k-delay Memory Capacity
MLP	Multi-Layer Perceptron
PCA	Principal Component Analysis
PoV	Proportion of Variance
RNN	Recurrent Neural Network
SCR	Simple Cyclic Reservoir

SCRJ	Simple Cyclic Reservoir with Jumps
tanh	Hyperbolic Tangent
var	Variance

Abstract

Effects of Network Weight Structure in Echo State Networks

Danny J. Wood

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy, 2020

Echo state networks (ESNs) are a type of recurrent neural model with a fixed internal weight structure and an adaptable readout trained using the network's hidden states as features. Since an ESN's weight structure is fixed (and often randomly generated), it is important that we understand how the behaviours which emerge from these networks are influenced by the design choices made in the network's construction. In this thesis, we examine the impact of several of these choices on the behaviour of ESNs. First, we examine the role of network weight structure in determining the memory capacity of ESNs. By drawing connections with results in control theory, we derive an expression for the memory capacity of a linear network in terms of structures within its weights. Next, we show that the previously reported phenomenon of deeper layers operating at slower 'time-scales' is exhibited even by linear networks, and we provide deeper insights into this behaviour by examining the sensitivity to perturbation and memory capacity of different layers. Finally, we examine the asymptotic behaviour of linear networks, and input-driven linear systems more generally. In the cases where the network is stable, we show properties which emerge from this stability, and construct tail bounds on the components of the hidden state when the network's input is perturbed by noise. In cases of instability, we construct bounds on the expectation of the hidden state's norm in the presence of noisy input.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

First and foremost, I would like to thank my supervisory team Ke Chen and Paul Glendinning, for their expertise and guidance. Without them, this thesis would not have been possible. I am grateful to Ke for fostering my interest in machine learning, and for his willingness to share his advice and technical expertise. Equally, I am thankful to Paul for his guidance, boundless enthusiasm, and his knack both for finding interesting questions and for offering clarifying perspectives on their answers.

I owe a great deal of thanks Andrew Webb for his willingness to listen to, and provide feedback on, even my most incoherent ideas, helping to shape them into something more sensible; and to Henry Reeve and Joe Mellor for sharing their knowledge and enthusiasm. I would also like to thank Jon Parkinson for his sound advice.

I feel lucky to have had many great people in my CDT cohort. Thanks to Alex, Amy, Cameron, Emily, Georgiana, Lara, Mike, Richard, Rob for sharing countless lunches, dinners and drinks. Spending my time in Manchester with such an amazing group of people has made it a wonderful experience, and I hope we remain friends long after we've all moved on to new and exciting things. Doing a PhD was also made immeasurably easier through sharing an office with many brilliant people, I am very grateful that I was able to spend my time in the research group in such great company. There are too many of you to name individually, but special thanks to Alessio Sarullo, Jon Crawford and Will Woof for proof-reading parts of this thesis.

In the wider school, I would like to thank John Latham and Giles Reger for sharing their experience and wisdom, and Gavin Donald, Susie Hymas and

Richard Ward in the student support office for always being helpful, friendly and cheerful in equal measure.

I would like to thank Mark and Emily for their sharing their home with me for the last two years. And thank you to my friends in Huddersfield for sharing in many memorable adventures, especially to Becky for her constant support and proofreading services. And thanks to Jax, for only barking at me when he felt I truly deserved it.

I am also very grateful to Gavin Brown for his help and support in preparing for my viva and final submission.

Thank you to the Engineering and Physical Sciences Research Council for funding my research through the the Centre for Doctoral Training Grant [EP/1038099/1].

Finally, thank you to my parents for their love, support and generosity.

Chapter 1

Introduction

It is a common aphorism that "it is difficult to make predictions, especially about the future".[1] While this is undoubtedly true, the ability to reason about the world and infer how a system will evolve given its current state and historical behaviour is an incredibly important one. In almost every scientific field, as well as in the practices of business and governance, the ability to make reasonable predictions about the future of a system given its past behaviour is vital in aiding our ability to both understand and influence complex systems.

In statistics and machine learning, a wide range of models have been developed to tackle the task of time series prediction. Classical statistical models for analysis of time series rely on the concept of auto-regression, using linear combinations of previous observations in order to predict the future values of the sequence. While these models can be very sophisticated and deal with a range of behaviours, the inherent linearity of the models ultimately restricts the kinds of sequences which they are able to successfully model.

Recently, the resurgence of neural networks has lead to a deeper exploration of recurrent neural models as a tool for processing complex time series. These models have proven to be very effective, despite the fact that they have several major drawbacks which are absent from neural models which operate in the feedforward paradigm. These problems are mainly concerned with the training phase, in which the model's parameters are learnt, with the recurrent nature of the model leading to many undesirable network behaviours.

[1]Though it's possible the sentiment was repeated by Danish physicist Niels Bohr—one of the many prominent figures to whom it is often attributed—it appears that the earliest written record attributes it to an unnamed member of Danish parliament, albeit with some claiming that its provenance is earlier as a Danish proverb [O'T13]

This difficulty lead to the emergence of reservoir computing, using the idea that a large untrained recurrent network can serve to generate a rich set of features, which a simple linear regression procedure is able to exploit to achieve good performance on time series prediction tasks. This idea was developed simultaneously for discrete time networks, in the form of Echo State Networks (ESNs) [Jae01] and continuous time networks in the form of Liquid State Machines (LSMs) [Maa+02]. It is the discrete time version, ESNs, which are of interest in this thesis.

1.1 Motivation

As interest in ESNs has grown, many efforts have been made to improve their design and attempt to overcome the shortcomings of previous network design strategies. It is important that this work of designing novel network architectures be performed in a reasoned and principled way, building on a strong foundation of theoretical knowledge regarding the impact of network design choices on the resulting network's properties and behaviour. In this thesis, we seek to build a deeper understanding of the impact of the choices made in the design of ESNs, with the hope that the ideas that we develop can be used to guide future researchers in their efforts to improve echo state network architectures.

A property of the network which will be of particular interest to us is its memory capacity (MC). The memory capacity of a network is, loosely speaking, the amount of information about the input history which can be inferred from the network's state. This is important in ESNs because, unlike other recurrent neural network paradigms, in reservoir computing the network's hidden state does not adapt to the patterns within the input sequence. Therefore, the network cannot learn to discriminate between relevant and irrelevant information, and the network must retain as much information as possible about the input history in order to ensure it is able to perform computations involving long-term dependencies between the input and desired output.

1.2 Research Questions

The primary question that we seek to answer in this thesis is a relatively simple one: *How does the structure of an Echo State Network affect its behaviour?*.

While this is an important question, it is a very broad one, and it is unlikely that a satisfactory answer to this question could be distilled into a single volume. Instead, we focus on sub-questions which can be meaningfully answered, and in this way attempt to contribute to the literature on the broader question. In particular, we attempt to answer the following, more specific, questions:

- How does the structure of a network's weights determine the amount of information that a network can contain? In particular, how can we infer the memory capacity of a linear network from properties of the network's weights?

- It is an oft-observed phenomenon that deeper layers of deep recurrent neural networks respond to inputs in ways that suggest that those layers operate at different time-scales. What are the causes of this phenomenon, and how does the structure of the network affect the ways in which it manifests?

- How does the structure of a network's weights determine its asymptotic behaviour? Particularly in the case of linear networks, how does the spectral structure of the network's recurrent weight matrix affect the long-term behaviour of the network and how is this influenced by the existence of noise in the network's input?

1.3 Contributions

We provide more detailed summaries of our contributions to each of the research questions above in the relevant chapters but, in broad strokes, the research contributions of this thesis are as follows:

- We illuminate the relationship between the memory capacity of linear networks and the notion of controllability in control theory. By leveraging this relationship, we show how we can analytically determine the memory capacity of a range of network structures, and that networks where weights are sampled from continuous distributions achieve the maximum possible memory capacity with probability one. Additionally, we find the same is true for deep networks under common random initialisation schemes.

- We provide insights into the relationship between the structure of deep echo state networks and the temporal behaviour of the layers. In particular, we conduct a series of experiments which demonstrate the effects of a layer's depth on how it recovers from perturbation of elements of the input sequence and on its memory capacity. Through empirical study, we provide novel insights into the nature of these phenomena.

- We provide a comprehensive overview of the relationship between the spectral structure of a linear network's recurrent weight matrix and the asymptotic network behaviour.

These contributions are the primary subjects of Chapters 4, 5 and 6, respectively.

1.4 Thesis Structure

The remainder of this thesis is organised as follows:

- Chapter 2 covers supervised learning and neural networks, providing context both for the types of problems ESNs are designed to solve, and the family of models to which they belong.
- Chapter 3 introduces ESNs and the related concepts required to understand the rest of the thesis.
- Chapter 4 examines how the structure of a network's weights affects the network's memory capacity.
- Chapter 5 examines the effects of depth in ESNs, particularly the causes of the phenomenon of different network layers 'operating at different time-scales'.
- Chapter 6 examines the asymptotic behaviour of linear networks in terms of the spectral radius of their recurrent weight matrix.
- Chapter 5 contains the thesis' conclusions and suggests directions for future work.

In addition to these chapters, the thesis contains four appendices. In Appendix A, we present some standard definitions and theorems from the literature; in particular, this appendix serves as a very brief overview of the standard results from linear algebra used throughout the thesis. Each of the three research chapters,

Chapters 4, 5 and 6, have an associated appendix (Appendices B, C and D, respectively) containing supplementary material (primarily proofs for intermediate lemmas which would disrupt the flow of the main text of the chapters).

Chapter 2

Background: Supervised Machine Learning and Neural Networks

Before discussing Echo State Networks (ESNs) themselves, we cover two prerequisite topics: firstly, we introduce supervised machine learning, and in particular sequence learning problems, this provides the motivation for the models discussed later in the thesis and describes the problems which they are designed to solve; secondly, we introduce artifical neural networks, and recurrent neural networks in particular, giving an overview of the class of models to which ESNs belong. A detailed introduction to ESNs, and related material is provided in Chapter 3.

2.1 Supervised Learning

Supervised learning is a machine learning paradigm in which a training set of input-target pairs is used to select, from a family of candidate models, the model which best describes the relationship between the inputs and the targets. More formally, we define an input domain $\mathbb{U}$, a target domain $\mathbb{T}$ and some joint distribution $\mathcal{D}$ over pairs (u, t) with $u \in \mathbb{U}$ and $t \in \mathbb{T}$. We would like to construct a function $f : \mathbb{U} \to \mathbb{T}$ which gives a good prediction of t, given that we know u, and that $(u, t) \sim \mathcal{D}$. In order to do this, we need to define exactly what is meant by a 'good prediction'. For this purpose, we choose a *loss function* $L : \mathbb{T} \times \mathbb{T} \to \mathbb{R}^+$, where $L(f(u), t)$ is a measure of the closeness of the prediction given by the model f for the input u to the target t. We would like $f(u)$ to generally be as close to t as possible for pairs (u, t) which have high likelihood under our distribution (i.e., likely input target pairs should yield a small loss). In particular, we would like to find a function for which the associated risk is as low as possible. Risk is measured using the risk function, defined as

$$R(f) \overset{\text{def}}{=} \mathbb{E}_{\mathcal{D}}[L(f(\mathbf{u}), \mathbf{y})].$$

In order to find such a function, we first select a family of models $\mathcal{F}$ (i.e, a set of candidate functions), which we believe to be expressive enough (that is to say, large and varied enough) such that at least one of its members will result in a suitably low risk. We then attempt to find the model f^* that gives the lowest risk amongst the models in $\mathcal{F}$; that is to say, we try to find

$$f^* = \arg\min_{f \in \mathcal{F}} R(f).$$

Unfortunately, the joint distribution $\mathcal{D}$ is not known, and therefore it is not possible to find the candidate model that minimises $R(\cdot)$ over the true distribution $\mathcal{D}$. Instead, we only have access to a training set of T input-target pairs sampled independently from the joint distribution, i.e., we have a set $S = \{(u_1, t_1), \ldots (u_T, t_T)\}$ with $(u_i, t_i) \sim \mathcal{D}$ for $i \in \{1, \ldots T\}$. As a proxy for finding the function in $\mathcal{F}$ that minimises the expectation of R over $\mathcal{D}$, we attempt to find the function that minimises the *empirical risk* over the set S. In other words, we look for $f^* \in \mathcal{F}$ which satisfies:

$$f^* \overset{\text{def}}{=} \arg\min_{f \in \mathcal{F}} \frac{1}{T} \sum_{i=1}^{T} L(f(u_i), t_i))$$

We search for f^* using a training algorithm: a procedure which takes our family of candidate models and training data set and selects, through some criterion, a function $f^\dagger \in \mathcal{F}$, which we believe to produce an empirical risk close to that of f^*. In some cases, such as linear regression, it is possible to find f^* through direct computation, though for more complex models, it is in general not possible to find f^* itself (or even to know if we have indeed found it).

Of course, since our training data is in the form of a finite set, an easy way to minimise the empirical risk is simply to have a function which acts as a lookup table for the values from $\mathbb{U}$ which occur in S, and returns the corresponding element in $\mathbb{T}$. With such a model, we have no guarantees that it will provide sensible outputs when the input is not in the training set: the model may not *generalise* well to unseen examples from the distribution. In order to prevent us from choosing such a model, care must be taken when selecting $\mathcal{F}$ to restrict the class of functions which it contains. One method of constructing $\mathcal{F}$ is to use a parameterised function with a large number of parameters, varying those parameters to give the functions which serve as elements of the set $\mathcal{F}$. For instance, we may construct a family of functions as

$$\mathcal{F} = \{f(\cdot;\Theta) \mid \Theta \in \mathbb{R}^N\},$$

where $N \in \mathbb{N}$ and $f : \mathbb{U} \times \mathbb{R}^N \to \mathbb{T}$. By limiting the number of parameters (i.e., by making N smaller), we limit the class of functions $\mathcal{F}$, and therefore prevent the models from being complex enough that they are able to simply memorise the dataset[1]. An additional way of preventing overfitting is to have the loss function penalise model complexity, biasing the choice of $f^\dagger$ to simpler models, even if those models have a higher empirical risk with respect to the original loss function. This process is known as *regularisation.* For instance, a common form of regularisation is penalising large values of the weights, encouraging parameters to have little/no influence on the function's value, unless having a larger value has a large effect on how well the function fits the training data. In this way, more complex models must justify themselves with a decrease in empirical risk commensurate with the degree of complexity of the function. A common form of this is L_2 regularisation, in which the regularisation term is proportional to the sum of the squares of the weights.

When it comes to evaluating how good our choice of $f^\dagger$ is, we once again run into the problem of not being able to distinguish between models which are fitting the distribution $\mathcal{D}$, and those which have just memorised the targets for the training data. In order to get an accurate measure of model performance, we generate a test set by sampling from $\mathcal{D}$ again, and calculate the empirical risk on that test set. Since we don't want our choice of $f^\dagger$ to be informed in any way by the test set, it is sometimes necessary to split the training set further. If, for instance, we have multiple candidate families of models, we might wish to use a training procedure on each of them, but withhold a set amount of training data which can be used to evaluate which model family has produced the best candidate,

[1]This is a slight simplification: it is not always true that fewer parameters mean a less expressive model, and with infinite precision computation it is possible to construct a parameterised function with a single parameter capable of perfectly fitting any training data set [Bou19]

and only calculate the empirical risk for the test set on that candidate model. In this case, the witheld set of training examples is referred to as a *validation* set.

2.1.1 Regression Problems

Broadly speaking, there are two categories of supervised learning problems of interest to machine learning researchers: classification and regression. In the former, $\mathbb{T}$ is a collection of discrete labels which offer a description of some aspect of examples which we would like to infer from the example's features. In the latter, the target is a scalar or vector value quantifying some characteristic of the data which can be inferred from the input features. It is the latter case with which we are most interested, in particular, we consider the case where $\mathbb{T} = \mathbb{R}$.

In this case, we wish for our loss function to serve as a measure of the distance between the prediction $f(u)$ and the target t. A common choice is to define distance using the Euclidean metric, which gives rise to the squared loss defined as

$$L(f(u), t) = (f(u) - t)^2.$$

For a model with parameters Θ, combination of the loss function with L_2 regularisation gives the empirical risk as

$$\frac{1}{T}\sum_{i=1}^{T}(f(u_i) - t_i)^2 + \lambda \cdot \|\Theta\|^2$$

where λ is the strength of the regularisation; for the unregularised case, we simply set $\lambda = 0$. This choice of loss function and regularisation are standard choices in the ESN literature (and beyond) but this does not mean that they are always the right choice: they come with a set of underlying assumptions about the the data and the model. Though these considerations are outside the scope of this thesis, it is worth noting that under a Bayesian framework, this loss function and regularisation term can be derived in a principled manner under the assumptions that the target is subject to additive Gaussian noise and that there is a Gaussian prior on the model weights [Bis06].

2.1.2 Sequence Learning Problems

Thus far we have specified the target domain $\mathbb{T}$, we now turn our attention to the input domain $\mathbb{U}$. A standard case in supervised learning is to have elements of $\mathbb{U}$ expressible as some finite length feature vector, e.g. $\mathbb{U} = \mathbb{R}^D$ for some D. While this is the canonical example for supervised machine learning, it is not always the most natural representation for the data sources that we may be interested in.

A problem which is of frequent interest in a wide variety of fields is predicting the future of a time series, given that time series' history. In particular, one might envisage a process starting at time $t = 0$, from which we are able to extract a sequence of observations $u_0, u_1 \ldots$, where each u_t is a scalar value, measuring some quantity of interest at time t. Given $u_0, \ldots u_t$, it would be useful to be able to select a model which is able to predict u_{t+k} for some positive integer k. In the paradigm of supervised learning, this gives a training set of input/target pairs of the form

$$\begin{aligned} ((u_0), u_{0+k}) \\ ((u_0, u_1), u_{1+k}) \\ ((u_0, u_1, u_2), u_{2+k}) \\ \vdots \end{aligned}$$

This presents a problem for standard machine learning methods, as the domain of the inputs is the set of all finite length sequences whose entries are in $\mathbb{R}$ (or $\mathbb{R}^D$ in the case of multi-dimensional inputs).

Classical auto-regressive time series prediction methods typically deal with this kind of data by selecting a fixed-length window (i.e, a fixed number of the most recently observed values) and constructing the model with inputs from this window as features (examples of such models are autoregression (AR), auto-regressive moving average (ARMA) and auto-regressive integrated moving average (ARIMA)). Though these models can be useful, they are limited by their assumption that the relationships between past and present inputs are fundamentally linear [Zha03].

Non-linear fixed-length window methods can be used in order to model more complex relationships between the most recent inputs and future values, and these have also been combined with the auto-regressive approach mentioned in the previous paragraph [Zha03]. Though they allow modelling of more complex non-linear relationships in the data, these approaches still have the problem that a fixed-length window must be selected from which to draw features.

A more natural method for modelling time series with neural networks is using recurrent neural networks (RNNs). In an RNN, the data is fed sequentially into the network, and the network state is updated to incoporate both the new information and information from the previous state of the network. In this way, the network contains a fixed-length representation of information from the entire history of the network. It is this family of models, to which our model of interest, the ESN, belongs. The remaining sections of this chapter are devoted to a description of this family of models, preparing the reader for the introduction of ESNs themselves in the next chapter.

Finally for this section, we note that though we frame the motivation for ESNs in terms of time series prediction—as this is the most common use case for ESNs—ESNs have also been used in other machine learning paradigms, such as sequence classification [TTC14].

2.2 Artificial Neural Networks

Artifical Neural Networks (ANNs) are a family of machine learning models which are based on the principle that information can be represented in a distributed manner by the collective state of a large number of number of components where the behaviour of each component is determined by simple rules. The name neural networks comes from the fact that the models are loosely inspired by the principles governing the workings of biological brains. Though the update rules for each component can be relatively simple, when taken in aggregate, they are capable of complex computation.

Though the history of ANNs is long and complex, with improvements and innovations being made over many decades (See [Sch15] for an in-depth review), in recent years ANNs have enjoyed a huge surge in popularity, particularly under the paradigm of *deep learning*. The development of methods for successfully training deeper models, where components are connected in a hierarchical manner, allowed networks to learn more complex functions of the input [HOT06], while improvements in computational power, as well as the ability to transfer computation to graphic processing units (GPUs), made it feasible to train larger networks and on larger datasets than was previously possible [Cir+11; KSH12].

In this section, we provide a brief primer on ANNs, starting with the most simple variety: Feedforward Neural Networks (FNNs), before discussing a common variant for dealing with sequential data: Recurrent Neural Networks (RNNs), of which ESNs are a sub-class.

2.2.1 Feedforward Networks

The simplest version ANN is the Feedforward Neural Network (FNN). The basic idea for this kind of network was first presented in [Ros58] as the preceptron, though the model we present here—the multilayer perceptron (MLP)—is significantly developed from Rosenblatt's, particularly in the use of hidden units. The MLP consists of three kinds of units: input units, hidden units and output units. In using neural networks, the goal is to approximate some unknown function $f : \mathbb{R}^D \to \mathbb{R}^O$. The input units consist of a D dimensional vector $\mathbf{u} \in \mathbb{R}^D$, and the output units are an O-dimensional vector $\mathbf{y} \in \mathbb{R}^O$. The hidden units form intermediate layers between the inputs and outputs. For an L layer network, we denote these hidden layers as $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(L)}$, with each $\mathbf{x}^{(i)} \in \mathbb{R}^{M_i}$ for some $M_i \in \mathbb{N}$. Here, as in the rest of the thesis, we will refer to a single entry in a hidden state vector as a hidden unit—with the number of hidden units of a network being equivalent to the sum of the lengths of the network's hidden state vectors. The values of the hidden states are determined by equations of the form

$$\begin{aligned}
\mathbf{x}^{(1)} &= \sigma(W_1\mathbf{u} + \mathbf{b}^{(l)}) \\
\mathbf{x}^{(i)} &= \sigma(W_i\mathbf{x}^{(i-1)} + \mathbf{b}^{(i)}) && \text{for } 2 \le i \le L \\
\mathbf{y} &= g(W_{L+1}\mathbf{x}^{(L)} + \mathbf{b}^{(L+1)}).
\end{aligned}$$

Here, σ is the activation function, commonly a function from the sigmoid family such as the hyperbolic tangent (tanh), or, alternatively, the rectifier or ReLU function[2]. The function g is chosen based on the particular kind of function that the network is being used to approximate. An example of an MLP is shown in Figure 2.1. The function computed by the network is determined by the weights and biases of the network, the W_is and the $\mathbf{b}^{(i)}$s, respectively. Together, these make up the *parameters* of the network. Here, each W_i is a matrix in $\mathbb{R}^{M_i \times M_{i-1}}$ (with the exceptions $W_1 \in \mathbb{R}^{M_1 \times D}$ and $W_{L+1} \in \mathbb{R}^{O \times M_L}$) and each $\mathbf{b}^{(i)}$ is a vector in $\mathbb{R}^{M_i}$ (with $\mathbf{b}^{(L+1)} \in \mathbb{R}^O$). For a given network architecture (i.e., a specific

[2]That is, for a hidden state vector $\mathbf{x}$, the element-wise function $\max(\mathbf{0}, \mathbf{x})$

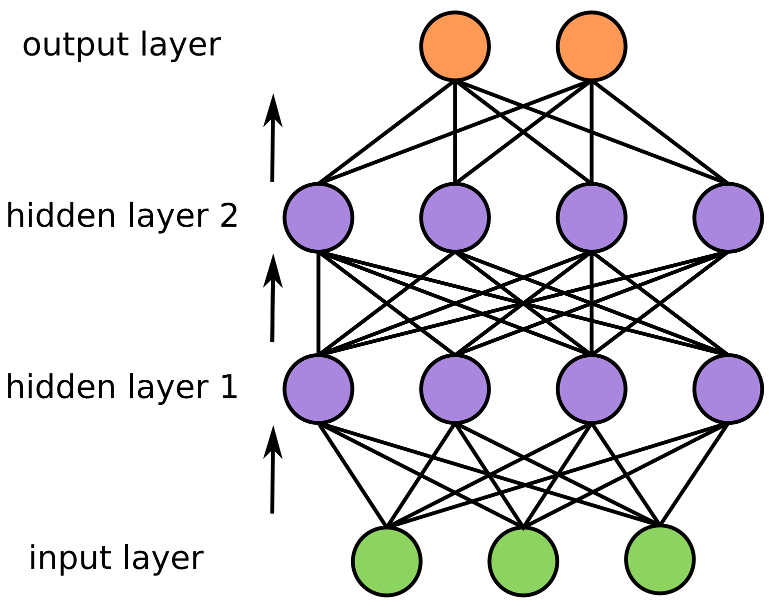


Figure 2.1: An MLP with three input units (green), two output units (orange) and two layers of four hidden units each (purple). The edges between nodes show dependencies between units, with the higher units in the image directly dependent on lower units. The bias term in the network definition is not represented in this image.

choice of activation functions, number of hidden layers and hidden layer sizes), varying these parameters allows us to select different members of the family $\mathcal{F}$ of candidate functions. The notation that we use here for our description of the network deviates slightly from the notation most frequently used to describe MLPs in the literature, we do this in order to make our notation as consistent with the notation for ESNs which we will introduce in the next chapter.

FNNs can approximate any smooth function on a finitely bound domain with arbitrary accuracy. This property, the *universal approximation property*, is even true of single layer networks, though the proof of this is non-constructive, and the number of hidden units required may be prohibitively large [Hor10]; it is for this reason that deep networks are preferred. It can be shown that for some classes of networks, that deep networks can compute some functions with exponentially fewer hidden units than their shallow counterparts [Has86; Mon+14; Tel16].

2.2.2 Recurrent Neural Networks

Feedforward neural networks impose the restriction that the examples and labels can both be represented by fixed-length vectors. However, for problems with a temporal structure, it is often desirable that the model can process sequences of arbitrary length. In order to accommodate this kind of input, it is possible to extend feedforward networks to include recurrent connections. The inclusion of these connections changes the role of the network from modelling a mapping between the input and output, to modelling a dynamical system. The input is fed into the network sequentially, with the hidden state of the network being updated in response to each input. The structure of the output of the network depends upon the exact nature of the task, though it is common to have an output computed at each time-step as a function of the network's hidden state (often concatenated with the network's input from that time-step).

A major advantage of recurrent networks over their feedforward counterparts is that, while feedforward networks can model any continuous function [Cyb89; Hor10], recurrent networks are able to approximate the dynamics of large classes of dynamical systems. In fact, the algorithmic fashion in which recurrent neural networks receive and process inputs allows them to compute any function that is computable by a Turing machine [SS92]. Interestingly, the converse is not true: it is possible for a recurrent network to simulate chaotic systems which are not computable by a Turing machine [Sie95]. However, harnessing the theoretical ability of RNNs for super-Turing computation requires that calculation is performed using infinite precision using real numbers [Sie95]. If we restrict ourselves to performing computation exclusively in the rational numbers, the ability to perform super-Turing computation disappears, and it becomes possible for a Turing machine to simulate any such RNN.

The literature contains a vast array of different recurrent network architectures but perhaps the most common network design is the Elman network [Elm90]. This network is such a commonly used baseline architecture that it is frequently referred to as a vanilla RNN [KJFF15; LB16; CSDS16; CPS18].

The basic structure of an Elman network is described by the equations

$$\begin{aligned} \mathbf{x}_t &= \sigma\left(W\mathbf{x}_{t-1} + V\mathbf{u}_t\right) \\ \mathbf{y}_t &= g\left(U\mathbf{x}_t\right) \end{aligned}$$

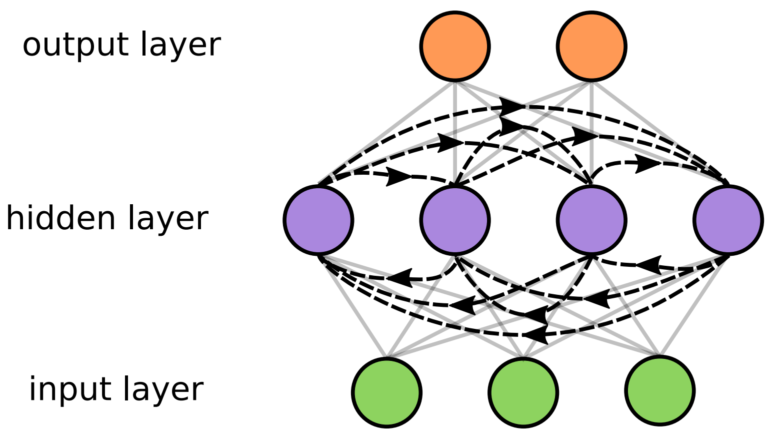


Figure 2.2: Diagram of a simple RNN. Connections through time are represented by dotted lines.

Here $\mathbf{x}_t \in \mathbb{R}^M$ is the hidden state at time $t \in \mathbb{N}$ and $\mathbf{u}_t \in \mathbb{R}^D$ and $\mathbf{y}_t \in \mathbb{R}^O$ are respectively the input and output vectors at time t. The network parameters $W \in \mathbb{R}^{M\times M}$, $V \in \mathbb{R}^{M\times D}$ and $U \in \mathbb{R}^{O\times M}$ are respectively the recurrent weight matrix, the input weight matrix and the output weight matrix. Like in the MLP, σ and g are element-wise activation functions. Figure 2.2 and 2.3 present two ways of visualising the Elman network. Visualising the network as in Figure 2.3 is referred to as *unrolling* the network.

2.2.3 Training Networks using Gradient-Based Optimisation

In order for a neural network to be useful for a given task, it is necessary to find parameters which allow it to approximate the desired mapping between the input and target spaces. In this section, we discuss the standard process of training networks. The difficulties in this process of determining a recurrent network's parameters is an important part of the motivation for the development of ESNs, and reservoir computing approaches more generally.

FNNs are most frequently trained using the backpropagation algorithm, introduced in [Rum85]. This algorithm consists of two parts: the forward pass and the backward pass. In the forward pass, an input vector is provided to the network, and the network output is calculated; the distance between the network output and the desired output is then calculated using the pre-selected loss function. In the backward pass, the derivative of the loss function with respect to each of the

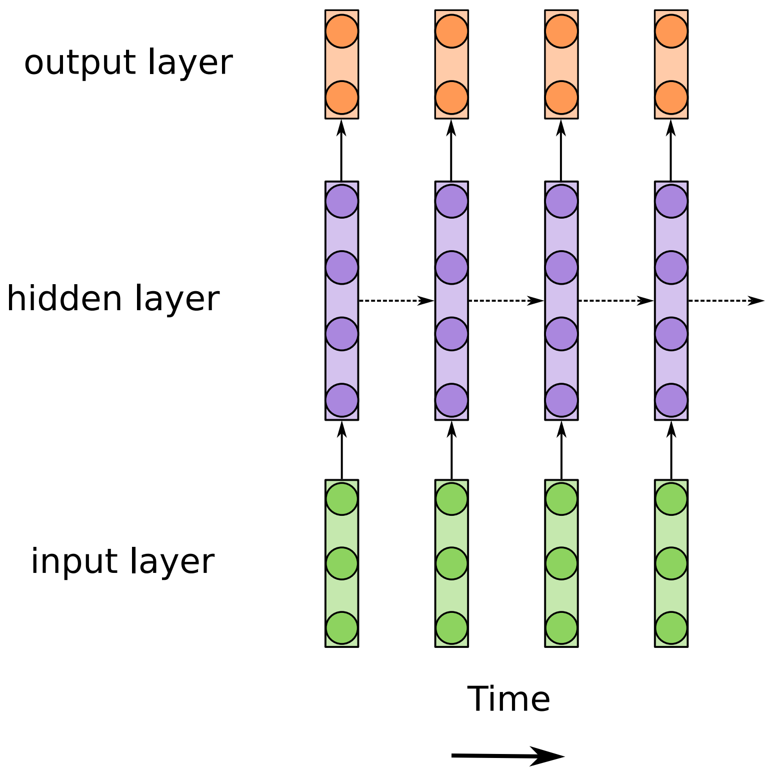


Figure 2.3: Diagram of an unrolled RNN. Arrows represent that each node inside the box the arrow is pointing to is dependent upon each node in the box the arrow is coming from. Feedforward connections are shown as solid lines and recurrent connections are shown as dotted lines.

network's parameters is calculated, and each parameter is updated proportionally to its derivative in the direction which reduces the loss. This process is performed either for a fixed number of iterations, or until the weights of the network converge. Typically, in each iteration, either a batch of training examples are processed, or the entire dataset, and the weights are updated with the average of the update vectors from each of the examples.

The introduction of recurrent connections significantly complicates the training process. Training is usually done through a fairly natural extension of backpropagation achieved by unrolling the network, as visualised in Figure 2.3. This method is referred to as backpropagation-through-time [Wer90]. Unrolling the network is essentially the same as treating the network as a feedforward network, but with an additional input at each layer. Here, the forward pass requires several applications of the same linear transformation matrix, the elements of the matrix are updated according to the sum of the contributions to the gradient from different numbers of these applications.

However, this iterated application of the recurrent weight matrix can lead to several issues in training. Most prominent among these are the vanishing and exploding gradient problems, first described in [BSF94]. The problem arises from the fact that the partial derivative of the error at time t with respect to parameters which effect the dynamics of the network is decomposable into a sum of products of partial derivatives between the error, network states and the network parameters. The nature of the products leads to an exponential weighting between terms in the sum, so the contributions to the gradient will be unreasonably skewed to either almost completely neglect the role of the distant past or even worse, have only the most distant past have any significant influence (these issues are the vanishing gradient problem and the exploding gradient problem, respectively). This is not the only problem, a recurrent network is essentially a parameterised dynamical system, in which bifurcations can occur, with small changes to the network weights having the possibility of causing major qualitative changes to the dynamics of the network [Doy92]. These unpredictable changes in network behaviour caused by small changes in parameters can have severe negative implications for the process of gradient descent [Doy92; PMB12].

Several approaches have been developed in order to avoid this problem in recurrent networks. One of the most popular approaches is the use of gated network units, which are constructed in such a way that the gradient updates are additive rather than multiplicative, removing the problem of exponential growth in gradient norms. Long Short Term Memory units (LSTMs) [HS97] and Gated Recurrent Units (GRUs) [Cho+14] are the most ubiquitous examples of this, though many variations on the idea exist [Gre+16]. This approach adds complexity to the structure of the network, but is extremely popular (and effective) when dealing with complex and multi-dimensional data with highly non-linear relationships between the inputs and the target (such as in language modelling).

Other approaches take more care with the gradient descent scheme, using more complex algorithms in order to mitigate the negative effects [PMB12; MS11], or else allow for the creation of more direct paths through the network, bypassing many time-steps via recurrent skip connections, i.e., direct connections from the state at time t to the state at time $t + k$ for some $k \geq 2$ [Zha+16].

We are interested in yet another alternative approach: reservoir computing (RC). Whereas the methods listed above deal with the problems inherent in training RNNs by adding complexity, and often incurring additional computational costs, reservoir computing can be thought of as taking the opposite approach: circumventing the problem by simplifying the training procedure. The next chapter of this thesis is dedicated to introducing RC model of interest to us: Echo State Networks.

Chapter 3

Background: Echo State Networks

In this chapter, we introduce our primary model of interest: Echo State Networks (ESNs). We first introduce the standard architecture for the model, before defining important properties of ESNs which will be of interest in later chapters, notably the Echo State Property (ESP) and Memory Capacity (MC). We finish the chapter by describing modifications and improvements to the standard network architecture which motivate some of the work later in the thesis.

3.1 Reservoir Computing

Reservoir Computing (RC) is a paradigm for machine learning models dealing with sequential data. In RC, a large but fixed-size hidden state is used to store a representation of the history of the inputs to the model up to the present time. The approach was developed independently in [Maa+02] and [Jae01], with the former focusing on continuous time systems in the form of Liquid State Machines, while the latter dealt with the discrete case, introducing Echo State Networks (ESNs).

In both cases the system is designed so that the hidden state contains a rich representation of the input, but the network's dynamics have a single stable state and satisfies a condition known as the *fading memory* or *echo state* property. This is the property that the effect of perturbations to the system fade over time. In [Maa+02], the analogy is made to the properties of a body of liquid. The surface of the liquid has a single stable state, which can be perturbed by external forces (e.g., pebbles dropping into a pool of water), causing ripples which propagate over the surface. By observing the state of the liquid at a given time, it is possible to infer where and when the events that caused the perturbations occurred, even though the system will—in the absence of further disruptions—asymptotically return to its original unperturbed state. In reservoir computing the hidden state is 'perturbed' by the input sequence, and by observing the hidden states, we are able to learn a mapping between these states and the target variable. Unlike the recurrent networks discussed in the previous chapter, the parameters dictating the structure of the hidden state are not learnt, and only the mapping between the states and the ouput is dependent upon the training data.

In terms of the supervised learning paradigm discussed in the previous chapter, this means that the class of candidate functions $\mathcal{F}$ is determined not just by hyperparameters like network size, but also by the parameters of the network (i.e., the network's weights), and selecting a candidate from $\mathcal{F}$ is done by choosing the appropriate parameters for the read-out weights.

Since its inception, RC has proven a popular paradigm for training recurrent neural networks. ESNs in particular have been shown to be a powerful tool in the field of non-linear system modelling and prediction, giving dramatic performance improvements over previous techniques [JH04]. This has lead to the successful application of ESNs to forecasting problems in diverse domains such as electrical power [DS12], weather forecasting [CST19] and finance [LYS09]. Using ESNs for more complex high-dimensional inputs has also proved fruitful, for example in speech recognition [SH07] and robotics [Ish+04].

3.2 Echo State Networks

We are now ready to give a formal definition of the ESN. We consider an ESN with M hidden units, driven by an input sequence $\{\mathbf{u}_t\}_{t=1}^{\infty}$ with $\mathbf{u}_t \in \mathbb{R}^D$. The sequence of hidden state vectors can be written as $\{\mathbf{x}\}_{t=1}^{\infty}$ with $\mathbf{x}_t \in \mathbb{R}^M$ and the sequence of output vectors as $\{\mathbf{y}_t\}_{t=1}^{\infty}$ with $\mathbf{y}_t \in \mathbb{R}^O$; these vectors are related to the input sequence by the update rules

$$\mathbf{x}_t = f(W\mathbf{x}_{t-1} + V\mathbf{u}_t) \tag{3.1}$$

$$\mathbf{y}_t = W^{out}\mathbf{x}_t. \tag{3.2}$$

Here, f is an element-wise 1-Lipschitz continuous function[1], such as tanh or the identity function. The matrices $W \in \mathbb{R}^{M\times M}$ and $V \in \mathbb{R}^{M\times D}$ remain fixed to the values to which they are initialised; it is this part of the network which we refer to as the reservoir. The exact initialisation scheme for these weights can vary greatly, but it is common for the elements of these matrices to be sampled independently from some carefully chosen distribution (the choice initialisation procedure will be discussed in greater detail in Section 3.4.1). In contrast with the input and feedforward weights, the weights of the output matrix $W^{out} \in \mathbb{R}^{O\times M}$ are determined by our chosen training procedure.

As a useful shorthand, we define $F(\mathbf{x},\mathbf{u}) \overset{\text{def}}{=} f(W\mathbf{x} + V\mathbf{u})$. With this, we are able to define the iterated update operator $G(\cdot,\cdot)$. This operator takes an initial state for the network $\mathbf{x}_0 \in \mathbb{R}^M$ and a finite-length input sequence $\{\mathbf{u}_1,\ldots,\mathbf{u}_T\}$ and returns the value of the hidden state of a network initialised with the initial state $\mathbf{x}_0$ and driven for T time-steps by the given input sequence. That is,

$$G(\mathbf{x}_0, \{\mathbf{u}_i\}_{i=1}^T) = F(F(\ldots F(\mathbf{x}_0,\mathbf{u}_1),\ldots),\mathbf{u}_{T-1}),\mathbf{u}_T).$$

We also introduce here a convention that we will rely on in later chapters of referring to a linear ESN by its parameters. That is to say, we will write (W,V) to refer to the linear ESN with recurrent weight matrix W and input weight matrix V. When the input a sequence of scalars rather than vectors (i.e, the case where $D = 1$), V becomes a vector, which we denote by $\mathbf{v}$ and we denote the linear network by $(W,\mathbf{v})$.

3.2.1 Training the Network

Arguably, the biggest advantage of ESNs over other recurrent neural networks is the ease with which they are trained. Though there are on-line techniques where the network is continually trained while receiving novel inputs, it is sufficient for our purposes to consider the off-line training regime. In this regime, the network is run for a training period of length T, the states generated by the network are collected into a matrix $X \in \mathbb{R}^{T\times M}$ (or alternatively both the state at time t and the input at that time are collected and concatenated into a single vector, giving $(X \in \mathbb{R}^{T\times(M+D)})$). Similarly the target values for the network during this period are concatenated into a matrix $Y \in \mathbb{R}^{T\times O}$, where $O \in \mathbb{N}$ is the dimension of

[1]A function $f : \mathbb{R} \to \mathbb{R}$ is 1-Lipschitz continuous if for all $x, y \in \mathbb{R}$, $|f(x) - f(y)| \leq |x - y|$

the target space. From these two matrices, we are able to determine the output weights which minimise the squared error between the network's states X and the desired output Y. The optimisation which computes such weights has a closed-form expression given by

$$W^{out} = Y^\intercal X(X^\intercal X)^{-1}$$

Frequently, ridge regression is used instead of linear regression. This is done by introducing a regularisation term, with the optimal weights now calculated as

$$W^{out} = Y^\intercal X(X^\intercal X + \beta I)^{-1}$$

where I is the $M \times M$ identity matrix and β is the *shrinkage parameter*. There are a couple of reasons why the introduction of the shrinkage coefficient is useful. Firstly, it allows the regression problem to be solved even if $X^\intercal X$ is singular. Secondly, a shrinkage parameter greater than zero penalises large weights, acting as a form of regularisation—in particular, L_2 regularisation.

When training a network, it is standard procedure to discard the initial states. This is due to the fact that the initial transient behaviour does not reflect the long-term dynamics of the network.

The fact that networks with untrained dynamics provide a set of features rich enough to give good predictive power with a simple linear regression is perhaps surprising, and the exact mechanisms behind it are not completely understood. However, there are several directions of research which shed some light on this phenomenon. It has been noted that networks can exhibit good performance on some tasks, even before the recurrent weights are trained, assuming that the initial random weights are small [TCB04]. This is caused by the *Markovian architectural bias* of such networks, the property that networks with similar recent histories will tend to have states which are tightly clustered together.

Recently, a series of papers [GO18; GO19] has examined the power of ESNs from the perspective of universality, showing that they can approximate large classes of causal time-invariant filters[2] with arbitrary accuracy. Perhaps surprisingly, these results extend even to ESNs with linear dynamics, as long as a non-linear (specifically, polynomial) transformation is applied to the states before training

[2]Roughly speaking, time-invariant filters are maps between infinite sequences which can be described by a mapping between a left-infinite input sequence and a scalar value.

the regression.

Using randomly initialised networks is related to the technique of random projections, in which data in a high-dimensional feature space is projected into a lower dimensional sub-space. This is a technique which has been fruitfully applied in other areas of machine learning and statistics. Random projections are a powerful tool, and have been shown to have desirable properties: random projections can be made from a high-dimensional space to a much lower-dimensional one while approximately preserving distances between a set of points with high-probability[JL84], and regressions performed on features generated using random projections offer competitive performance compared to far more computationally-intensive models which they approximate [RR08; RR09]. The complexity added by the iterated application of the recurrent weight matrix in ESNs, along with a non-linear activation function means a direct application of the theory developed in these works to ESNs is not possible, but they give a flavour of the power of random projections as a tool in machine learning.

3.2.2 The Echo State Property

In order to ensure stability in the hidden state, we desire that ESNs satisfy the Echo State Property (ESP). Roughly speaking, the ESP means that the network's state is asymptotically independent of the network's initial conditions. This ensures that the network's state is most heavily influenced by the recent past, and that inputs cannot cause the network's state to become permanently trapped in a restricted region of the state space for all possible future inputs. Formally, we state the echo state property with the following definition.

Definition 3.2.1 (Echo State Property). *Let $G(\cdot,\cdot)$ be the iterated update operator describing the dynamics of some ESN accepting D-dimensional inputs. We say the ESN has the echo state property if there exists a sequence $\{\delta_i\}_{i=1}^{\infty}$ with $\lim_{i\to\infty}\delta_i = 0$ such that for all right-infinite input sequences $\{\mathbf{u}_j\}_{j=1}^{\infty}$ where each $\mathbf{u}_j$ satisfies $\mathbf{u}_j \in U \subset \mathbb{R}^D$ for the compact set U and for all $\mathbf{x}_0, \mathbf{x}'_0 \in A \subset \mathbb{R}^M$, where A is a compact subset of $\mathbb{R}^M$ containing all permissible network states, $d(G(\mathbf{x}_0, \{\mathbf{u}_j\}_{j=1}^{t}), G(\mathbf{x}'_0, \{\mathbf{u}_j\}_{j=1}^{t})) < \delta_t$, where $d(\cdot,\cdot)$ is the Euclidean distance function.*

Along with this definition, Jaeger's original paper [Jae01] provides two results giving sufficient and necessary conditions for the property to occur. These conditions are stated in terms of the *spectral radius* and the *operator norm* of the matrix W. Before stating the conditions, we define these terms.

Definition 3.2.2 (Spectral Radius). *For a matrix A, the spectral radius, $\rho(A)$ is defined as the maximum absolute value amongst the eigenvalues of A, that is,*

$$\rho(A) \overset{\text{def}}{=} \max\left(|\lambda| \,:\, \lambda \in S(A)\right),$$

where $S(A)$ is the set of eigenvalues of A.

Definition 3.2.3 (Operator Norm). *For a matrix $A \in \mathbb{R}^{M\times N}$, it's operator norm, $\|A\|$ is defined as*

$$\|A\| \overset{\text{def}}{=} \max_{\mathbf{x}\in\mathbb{R}^N} \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|},$$

where the norm used on the right-hand side is the vector 2-norm (i.e., $\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{N} |(\mathbf{x})_i|^2}$)

With these definitions, we can state the necessary condition for the ESP as:

Proposition 3.2.1. *For an ESN with update rule given in Equation 3.1 and 1-Lipschitz activation function f, in order for the ESP to hold it is a necessary condition that $\rho(W) < 1$.*

The sufficient condition is as follows.

Proposition 3.2.2. *For an ESN with update rule given in Equation 3.1 and 1-Lipschitz activation function f, the ESP holds if $\|W\| < 1$.*

It is interesting to note that these conditions are independent of the structure of V. For the sufficient condition, the proof allows for arbitrary inputs and proof of the necessary condition assumes only that the zero vector is a valid input. Making stronger assumptions about V and the statistical properties of the input sequence yields a different set of conditions for the ESP, but this is beyond the scope of this work [MJ13].

For many initialisation schemes for the network weights, there is a significant gap between the sufficient condition and necessary condition given by these two propositions. As an example, take the initialisation schemes where W has elements drawn from some zero-mean distribution with variance $\frac{\sigma^2}{M}$ for some $\sigma^2 > 0$. The asymptotic behaviour of the network as $M \to \infty$ is agnostic to the choice of distribution, as long as the fourth moment of the distribution is bounded [BY86]. In particular, we have that as $M \to \infty$, the spectral radius of W tends to σ^2 [BY86] almost surely. However we also have that the largest singular value tends to $2 \cdot \sigma^2$ almost surely [YBK88]. Due to the size of this gap between the sufficient condition and the necessary one, much effort has been invested in attempting to find less restrictive sufficient conditions. In particular [BY06] shows that it is sufficient for the norm of W to be less than unity in any one of a family of norms, of which the operator norm is a member. This idea was reformulated in [YJK12] to be expressed in terms of the Schur stability of W. Though these results improve upon the previous sufficient condition in [Jae02b], empirical evidence suggests that the gap between the sufficient and necessary conditions in randomly initialised networks remains large in practice [ZMW12].

The existence of this gap between the sufficient and necessary conditions causes problems in the principled application of ESNs to practical problems. The sufficient condition is often too restrictive, giving recurrent weight matrices where the weights are too small and leading to networks which are incapable of retaining information over long time periods. Meanwhile, it can be tempting to treat the necessary condition as a sufficient one: large networks satisfying the necessary condition often have properties one would expect of networks satisfying the echo state property [Cal+13], even if it isn't provably the case. In [ZMW12], more support is given to the idea that the necessary condition is likely to be sufficient for large enough random recurrent weight matrices: it is shown that a spectral radius less than one means that the probability that the distance between two randomly chosen states is increased by a network update is exponentially small in the size of the network when the weights are chosen in an i.i.d. manner [ZMW12]. This has lead to some confusion amongst researchers, with some conflating the necessary condition with the sufficent one, while other researchers warn of the dangers of such a conflation [YJK12]. The confusion is compounded further by the practical advice that one should try spectral radii larger than one in conducting a hyperparameter search [Luk12], as such configurations can offer better performance on certain tasks.

Other efforts have reframed the echo state property to be dependent upon the statistical properties of the inputs, loosening the restrictions discussed above for certain kinds of inputs. [MJ13] notes that the standard conditions do not consider the role of a saturating non-linearity, which creates a compressive effect drawing large pre-activation values much closer together than it would small pre-activation values, and therefore if the inputs are sufficiently large a new looser sufficient condition can be constructed.

3.3 Memory Capacity

The hidden state of an ESN is often thought of as having two roles: storing a representation of the network's input history and performing non-linear computations on that representation. In order to measure the network's ability to perform the first of these tasks, the amount of information stored within the state of the network is often quantified by the network's Memory Capacity (MC). Memory capacity is a measure that was introduced in [Jae02b] to measure the information stored in the network. Though we refer to this quantity as memory capacity or MC, we note that it is also common in the literature to refer to it as Short Term Memory (STM).

The memory capacity of an ESN is a measure of how well the network retains information about the past inputs. The k-delay memory capacity of a network is the square of the Pearson correlation between the input at time $t-k$ and the optimal linear reconstruction of that input from the hidden state vector at time t. If we consider the network $(W, \mathbf{v})$, the k-delay memory capacity quantifies the amount of information that can be captured from the networks hidden state $\mathbf{x}_t$ about the input k steps in the past, and the total memory capacity is the sum of the k-delay capacities over the positive integers. Typically, we are concerned with memory in the worst-case scenario: the scenario where the input sequence is random, with no correlation between the inputs at different times. While this is not a realistic use case, correlations between inputs allow the network to recall more inputs with more clarity, and therefore the performance in the uncorrelated case gives a lower bound on the network's performance on correlated inputs. Throughout the rest of this thesis, when talking about memory capacity we will specifically be discussing the memory capacity under this worst-case scenario; that is, we assume the network is driven by a sequence of random scalar inputs

$\{u_t\}_{t=-\infty}^{\infty}$ where each u_t is zero mean and has the same finite variance. If we define

$$\mathbf{z}_t \overset{\text{def}}{=} \begin{pmatrix} u_t \\ \mathbf{x}_t \end{pmatrix}$$

and $\mathbf{w}_k \overset{\text{def}}{=} \arg\min_{\hat{\mathbf{w}}_k} \mathbb{E}\left[(\hat{\mathbf{w}}_k^\intercal \mathbf{z}_t - u_{t-k})^2\right]$, then the k-delay memory capacity of the network is defined by the expression

$$\mathrm{MC}_k(W, \mathbf{v}) \overset{\text{def}}{=} \frac{\mathrm{cov}^2(\mathbf{w}_k^\intercal \mathbf{z}_t, u_{t-k})}{\mathrm{var}(\mathbf{w}_k^\intercal \mathbf{z}_t) \cdot \mathrm{var}(u_t)},$$

where cov and var denote the covariance and variance, respectively. Here, the expectation is with respect to the distribution over input sequences. Note that we assume that the network has been running since $t = -\infty$, so the value of MC_k is the same, no matter what (finite) value of t we consider when taking expectations. We refer to the sequence of values of MC_k for $k \in \mathbb{N}$ as the *memory capacity curve* of the system. The total memory capacity of the system (MC), is defined as the infinite sum

$$\mathrm{MC}(W, \mathbf{v}) \overset{\text{def}}{=} \sum_{k=1}^{\infty} \mathrm{MC}_k(W, \mathbf{v}). \tag{3.3}$$

When memory capacity of a system is evaluated numerically, quantities are calculated by averaging over time-steps in a single run of the network. By convention, this run is done on a long sequence of randomly generated scalar values over the interval $[-1, 1]$ [Jae02b; FBG16]. It is worth noting that it is not uncommon in the ESN literature to measure memory capacity by performing regressions on $\mathbf{x}_t$, rather than $\mathbf{z}_t$. We prefer $\mathbf{z}_t$, partly due to the fact that it is the original definition as given in [Jae02b], and partly because it has properties which will be useful to us in Chapter 4. Since this difference in definition corresponds to adding a single feature to the data in the regression problem, the difference between the two definitions in terms of required computational resources is minimal.

It was shown in [Jae02b] that the memory capacity of a network with M hidden units is at most M. The same paper also provided the following proposition describing the situation in which this maximum memory capacity could be achieved.

Proposition 3.3.1 ([Jae02b], Proposition 4)**.** *For a linear ESN* $(W, \mathbf{v})$*, define* *the matrix* $M_{W,\mathbf{v}} \overset{\text{def}}{=} \left(\, W\mathbf{v} \, \middle| \, W^2\mathbf{v} \, \middle| \, \cdots \, \middle| \, W^M\mathbf{v} \, \right)$*, i.e.,* $M_{W,\mathbf{v}}$ *is the* $M \times M$ *matrix whose* i*th column is the vector* $W^i\mathbf{v}$*, where* W^i *is* W *to the* i*th power. The memory capacity of a linear ESN is* M *if and only if the matrix* $M_{W,\mathbf{v}}$ *has full rank.*

When determining memory capacity via empirical experiment, since it is not possible to compute an infinite sum in Equation 3.3, a maximum value k_{max} is selected, and the truncated sum calculated from 1 to k_{max}. As long as a sufficiently large value of k_{max} is chosen (usually at least the number of hidden units in the network), the effects of using this approximation are minimal, since the memory capacity is at most M, and plotting MC_k usually reveals a plateau structure where there exists some $1 \leq k' \leq M$, such $MC_k \approx 1$ for $k \leq k'$, then dropping off quickly to zero above that value.

Though memory capacity is the most popular measure in the echo state literature, it is not the only method by which the memory of an ESN can be quantified. The Fisher memory curve offers another perspective on the memory of an input-driven system, quantifying the senstivity of the system to perturbations in its input history [GHS08]— though it can be shown in linear networks, the Fisher memory curve is related to the memory capacity curve by a result presented in [TR13]. Additional measures were investigated in [Boe+11], using an information theoretic approach to examine how much information was stored in each unit of the reservoir about the network's state history. Interestingly, these additional measures, along with memory capacity, are all empirically shown to be maximised as the spectral radius of W approaches one.

It is important to note that the memory capacity quantifies only the network's ability to retain information, but says nothing of its ability to process that information in order that meaningful computations on the input might be extracted. There have been attempts to extend memory capacity to give a broader computational capacity measure of the network. Particularly notable for its principled approach is [Dam+12], in which the ability of the network to approximate Legendre polynomials of its inputs is systematically quantified.

3.4 Reservoir Initialisation Methods

In this section, we discuss some strategies for initialising the reservoir of an ESN (i.e., the recurrent weight matrix and the input weights). Since these weights are fixed before the network is trained, choosing a suitable initialisation is vital to the network's ability to perform well on a given task. Here, we examine two particular kinds of strategies: random initialisations and deterministic reservoir construction.

3.4.1 Random Reservoir Constructions

The original description of the ESN called for the recurrent weight matrix to be a large sparse randomly generated matrix, claiming that this would give rise to a rich set of dynamics [Jae01]. Though many alternative methods of initialisation have been suggested over the years, networks where matrix entries are drawn independently from some pre-chosen distribution are still extremely common. Likewise, the method suggested in [Jae01] for initialising the input weights—sampling them from a uniform distribution over an interval centred at zero—is still common practice. For the input weights, the size of the interval determines the input scaling, and therefore how large a compressive effect the activation function will have in non-linear networks.

The sparsity constraint on W was initially assumed to be important in ensuring the network would function optimally—with sparsity promoting diversity of features by restricting communication between different parts of the network—but this has proved to make little difference in practice. On the other hand, it is pointed out in [Luk12], that a potential advantage of sparsity comes not from an improvement in performance on the learning task, but on the time complexity of training and inference. In order to get this improvement, rather than having sparsity as a fixed ratio of network size, the number of outgoing connections from each unit is fixed instead; this kind of sparsity constraint can be used to reduce the computational complexity of the matrix operations. For sparse networks, the non-zero elements of the recurrent weight matrix can be chosen either from a uniform distribution with zero mean or from the two element set $\{-a, a\}$ for some positive real number a.

The spectral radius of a large random matrix is determined asymptotically almost surely by the variance of its entries, and similarly for the operator norm [BY86; YBK88]. These results allow confidence in achieving a given network behaviour, even without the computationally expensive step of computing the SVD or eigendecomposition in order to scale the matrix to give the desired value. However, as discussed in the previous section, when generating random matrices in an i.i.d. manner, we encounter a problem due to the gap between the variance needed to satisfy the sufficient condition for the ESP and the variance required for the necessary condition [ZMW12]. This problem can be resolved while still allowing construction of a random network by using symmetric matrices, e.g. a Wigner ensemble, as in [Tin17]. This causes the spectral radius and operator norm to coincide, and therefore their shared value being below one is enough to guarantee the echo state property. The same effect is achieved in [FG17], albeit with an alternative approach: a random recurrent weight matrix is generated, then an orthogonalisation or orthonormalisation procedure is applied. Despite these, and other works [WLS04] on orthogonal recurrent weight matrix initialisations, how well these reservoir constructions work in practical applications is a question that has not been explored.

Another approach to generating random matrices with more structure is by constructing a *decoupled reservoir*, as in [ZW08]. In a decoupled reservoir, hidden units exist in pairs or in isolation, with each hidden unit receiving recurrent updates from only itself or itself and the unit it is paired with. It is shown in [ZW08] that such networks have the same approximation ability as a network with a dense recurrent weight matrix when the identity is used as an activation function, and such a network can perform better than the standard ESN on some tasks. In order to generate a decoupled reservoir, the authors suggest sampling values from the unit disk in the complex plane and using those values to define the coupling strengths of pairs of hidden units. Asymptotically, this gives a matrix with the same density of eigenvalues on the complex plane as if the elements of W were sampled i.i.d.. This idea, constructing a recurrent weight matrix by first designing a set of desirable spectral characteristics and building a recurrent weight matrix to have those characteristics, was also explored in [OXP07], using the rational canonical form of the matrix [BW89], though in this case, an unsupervised learning algorithm is also applied to adapt the weights to the input sequence.

Another approach is to generate reservoirs algorithmically, but with steps in the algorithm having probabilistic rules. In [DZ07], a method of constructing reservoirs is suggested in which a sparse recurrent weight matrix is generated in which hidden units are clustered into groups, with units in the same group having non-zero weights with high-probability, and few non-zero weights connecting nodes from distinct groups. Though the global structure of these matrices is determined algorithmically, the local connections are chosen randomly. The approach in [DZ07] generates weight matrices which are small-world (i.e., the distance between nodes is logarithmic in network size) and scale-free (the distribution of number of connections that nodes have obeys a power law). The combination of these two properties appears to have beneficial effects on network performance, whereas networks with only the small world property appear to only do better than i.i.d. entry networks when the number of hidden units which are connected to the inputs and outputs are restricted [Kaw+17; KPA19].

3.4.2 Deterministic Reservoir Construction

Early literature on ESNs would frequently place an emphasis on the importance on choosing a reservoir initialization which maximised the richness of the representation of the hidden state. It was not however, made explicit exactly what richness of representation meant, or how best it could be achieved. While increasing the complexity of networks could potentially improve the network in this regard, a pair of papers [RT11a; Rod12] asked the opposite question: How much can the network structure be simplified while still giving rise to competitive models? In these two papers, three main reservoir structures were proposed: the Delay Line Reservoir (DLR), the Simple Cycle Reservoir (SCR) and a Simple Cycle Reservoir with Jumps (SCRJ)[3].

In the DLR, the hidden units form a chain, the input is received by only the first unit, and successive units are updated based solely on the state of the previous unit at the previous time-step, as seen in Figure 3.1a. This can also be augmented with feedback connections, as seen in Figure 3.1b. SCR is similar to DLR, but with the units forming a ring rather than a chain, as seen in 3.1c. In the SCRJ this ring structure is further augmented by having periodic connections between non-adjacent units in the ring, as visualised in Figure 3.1e. The SCRJ can be extended to have jumps of different sizes, as in the Cycle Reservoir with Hierarchical Jumps (CRHJ), shown in Figure 3.1f. While the SCR achieves close to the same performance as a typical ESN a range of benchmark tasks, the SCRJ has been shown to be able to frequently exceed it.

The recurrent weight matrices of these networks are deterministic and highly structured, but the input weight matrices need to be in some sense unstructured in order for the network's performance to not deteriorate. While these weights

[3]It is also worth noting the existence of the earlier, exploratory work by [FE05] in this direction, examining the effects of a strictly diagonal recurrent weight matrix.

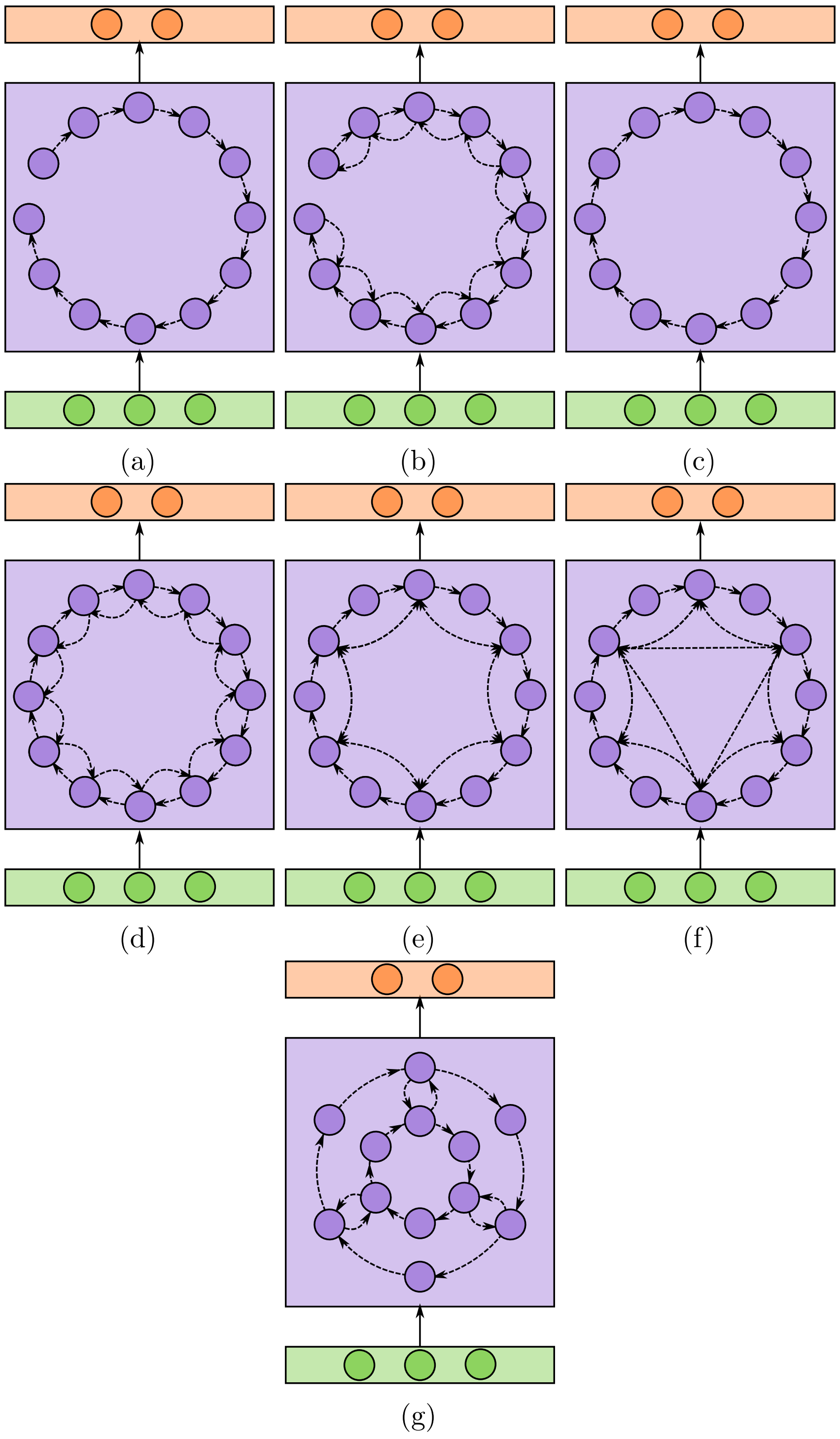


Figure 3.1: Reservoir topologies for various reservoirs introduced in [RT11a] and [Rod12]. The topologies are: (a) Delay Line Reservoir, (b) Delay Line Reservoir with Feedback, (c) Simple Cycle Reservoir, (d) Adjacent-feedback Loop Reservoir (e) Cycle Reservoir with Jumps, (f) Cyclic Reservoir with Hierarchical Jumps. (g) Concentric Echo State Network.

can be chosen in a deterministic way, e.g. by determining the weights based on successive digits of an irrational constant, the networks still rely on a degree of pseudo-randomness to generate useful network dynamics. This is in order to break the symmetry that could otherwise be present in the networks, which would cause hidden units in distinct regions of the network to contain identical information.

These ideas have been built upon in several papers by other researchers. In [Sun+12], the SCR is modified so that each unit in the hidden state is connected to both the next and previous unit in the ring, producing the Adjacent-feedback Loop Reservoir (ALR) as shown in Figure 3.1d, and another recent paper built further explored the space of possible models with the Concentric Echo State Network (CESN), which builds upon the SCR and CRJ with the inclusion of multiple interconnected cycles [BB18], an example of this network structure is shown in Figure 3.1g. The more elaborate structures proposed in [BB18] trade away some of the simplicity of the models in [RT11a] [Rod12], but appear to offer consistently better performance across a range of tasks and network sizes. These results suggest that further exploration of similar reservoir topologies is likely to yield further improvements. Though these improvements often add complexity to the network design, they still offer a deterministic network design which can be described and exactly replicated with greater ease than randomly generated networks. Additionally, the simple structure of the network allows for reservoir adaptation techniques which apply specifically to these networks, and can be used in order to efficiently find reservoir constructions which provide the best performance for a specific task (e.g, [YN15], [WJY15], [TTC14]).

3.5 Deep and Modular Structures

A key property of ESNs is that the network state at a given time serves as a rich representation of the input history up to that point. As such, a range of strategies have been developed to promote diversity in the features of the hidden state. One such strategy is to modularise the hidden state, with different parts of the state being subject to different update rules, and therefore containing different representations of the input history. In this section, we examine some of these strategies, focusing particularly on deep networks, but also briefly discussing other methods of modularising the network.

3.5.1 Deep Networks

In the feedforward case, the strategy of constructing networks in a hierarchical manner was motivated by the idea that composing hidden states in such a way allows the representational power of the network to grow exponentially with the number of layers [Mon+14], and deeper layers of the network are able to learn more abstract representations of the input [Ben09]. Since the success of [KSH12] on the task of image classification, deep learning methods have come to be a dominant method in machine learning—and since the qualities of deep networks that we've just described are equally desirable when dealing with sequential data, much effort has been made to construct deep recurrent networks.

Early precursors of modern deep recurrent architectures include the work of [Sch92], in which networks were constructed in order to contain reduced length descriptions of their inputs, with the first layer encoding as much information about the input sequence as possible, and successive layers of the network used to encode information not successfully captured by the previous layers. Another notable early work [EB96] introduced the idea of separating the hidden state into several components with different dependencies and update frequencies. The networks in [EB96] are a precursor to today's deep recurrent networks, and though the networks are not on the same scale as modern architectures in terms of the size of the hidden state, the paper contains examples of features such as skip-connections and hierarchical states which would become popular many years later. These models serve as early examples of the most common method of adding depth to recurrent networks: hidden state stacking. In a model with stacked hidden states, multiple hidden state vectors exist at each time-step, with the state held in a layer determined by an update rule combining the value of that layer's state vector at the previous time-step and the current value of the state of the layer directly beneath it. This variety of deep network has proven to be successful in tackling problems in a wide variety of domains, such as time series classification for medical diagnosis [Lip+15], handwriting recognition [Gra12], language modelling [KJFF15], and as part of a pipeline for image classification in video [Ng+15]. Outside of reservoir computing, these deep networks most commonly use layers of Long Short Term Memory (LSTM) units for these tasks, but other layer structures such as simple sigmoid layers or Gated Recurrent Units (GRUs) [Cho+14] can just as easily be applied.

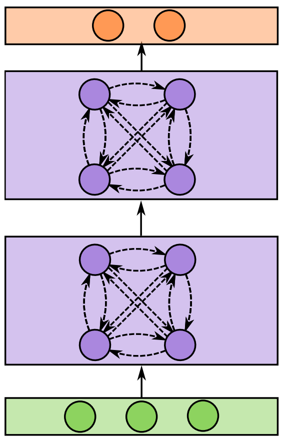


Figure 3.2: Diagram of two layer DeepESN with two hidden layers of four units each.

State stacking can also be applied to ESNs as a hierarchical method of modularising the network structure, producing the Deep Echo State Network (DeepESN), as shown in Figure 3.2. We consider a DeepESN with L layers of M hidden units each. The first layer of the network is defined in a manner similar to the single-layer ESN, with the update equation

$$\mathbf{x}_t^{(1)} = f(W_1 \mathbf{x}_{t-1}^{(1)} + V_1 \mathbf{u}_t)$$

where $\mathbf{x}_t^{(1)} \in \mathbb{R}^M$ is the hidden state of the first layer at time t, $W_1 \in \mathbb{R}^{M \times M}$ is the recurrent weight matrix for the first layer of the network, $\mathbf{u}_t \in \mathbb{R}^D$ is the input at time t and $V_1 \in \mathbb{R}^{M \times D}$ is the input weight matrix (we may also write $\mathbf{v}_1$ if $D = 1$). Subsequent layers of the network are defined iteratively for $2 \leq i \leq L$ with the update equations

$$\mathbf{x}_t^{(i)} = f(W_i \mathbf{x}_{t-1}^{(i)} + V_i \mathbf{x}_t^{(i-1)}),$$

where $W_i \in \mathbb{R}^{M \times M}$ and $V_i \in \mathbb{R}^{M \times M}$ are the recurrent weight matrix for the ith layer and the feedforward weight matrix for the ith layer, respectively. In the DeepESN, each $\mathbf{x}_t^{(l)}$ can be calculated as a function from state of the previous layer ($\mathbf{x}_t^{(l-1)}$) and the state of the current layer at the previous time-step ($\mathbf{x}_{t-1}^{(l)}$). A visualisation of these dependencies is shown in Figure 3.3.

As reported in [HS13], layers of a deep recurrent network with stacked hidden states differ in regards to how their response to inputs varies as greater and greater

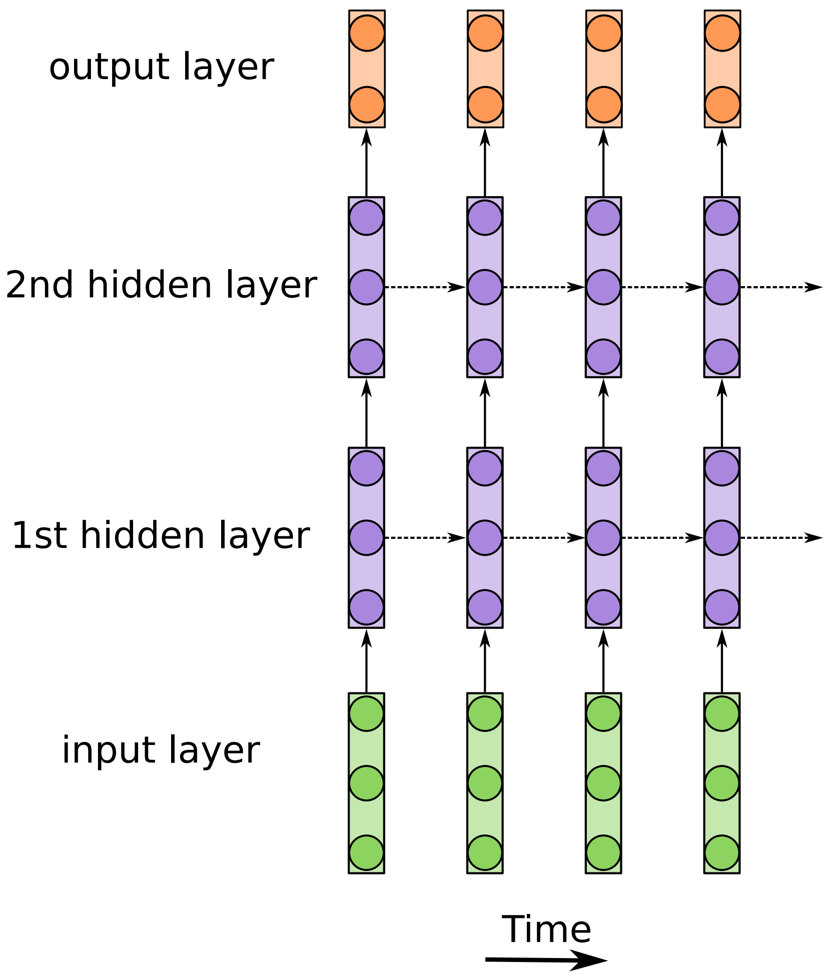


Figure 3.3: Diagram of an unrolled DeepESN with two hidden layers of three units each.

time-lags are introduced between the relevant input and when the network's state is observed. This phenomenon is referred to as different layers of the network operating on *different time-scales*. This phenomenon has also been shown to exist in untrained networks, such as DeepESNs [GM16]. Additionally, a series of papers by Gallicchio et al have provided an appraisal of many other desirable properties of DeepESNs, including in terms of the echo state property [GM17], memory capacity [Gal18], frequency response[GMP19] and Lyapunov exponents [GMS18].

Other methods of implementing depth in ESNs have also been explored, notable examples include [MHW17], in which the weight matrices which parameterise the transformations of data at each layer are identical, and the models presented in [Car+18], which experimented with various reservoir topologies as methods of incorporating depth into the network.

The method most commonly used in training output weights in deep ESNs is to use as features the hidden states at every layer. This unfortunately introduces the problem that as the depth of the network increases so does the number of features on which the regression is performed. In turn, this increases the number of training data points required in order to successfully train the network. As a remedy to this, [MSC17] introduces encoding layers between the high-dimensional reservoir at each layer, performing the regression on a concatenation of these lower dimensional encodings. In [Liu+18], a simpler approach is used. Rather than concatenating the states, regression is performed on each layer separately, and the outputs from each layer ensembled by taking the arithmetic mean.

It is worth noting that in recurrent networks, there is no single definition of depth, and that there are a multitude of methods by which depth can be implemented in recurrent architectures. [Pas+13] and [Zha+16] represent two attempts to systematise the study of deep recurrent networks. The former examines the various places in which 'depth' can be added to a network, discussing the effects that such depth could have on computation, and the latter attempts to quantify the depth of such networks with the introduction of architectural complexity measures. While the intent of both papers was the study of recurrent networks subjected to backpropagation to minimise a loss function, they suggest a wide and largely unexplored, search space for echo state network models.

3.5.2 Other Modularisation Strategies

Though in this section we have primarily focused on depth as a method of modularising networks, it is worth noting that other methods of modularising a network have also been attempted, with the goal of fostering diverse representations amongst the modules.

Perhaps the most extreme approach to modularising an ESN is to have distinct reservoir components with non-interacting dynamics, with the only interaction between different parts of the network happening in the combining of the outputs, this essentially leads to an ensemble of independently operating networks. This approach was demonstrated in [SL09], showing favourable performance of ensembles of ESNs compared to ensembles of multi-layer perceptrons on sequence prediction tasks.

In [XYH07], a modular structure was examined whereby the network is constructed to have distinct sub-reservoirs, but these reservoirs are coupled together using *lateral inhibition* to ensure de-correlation between the dynamics of the separate reservoirs and therefore improve their collective performance. In [Qia+17], a different approach is used, progressively constructing a reservoir in a modular fashion, until some stopping criterion, based on the task-specific performance of the reservoir, is met.

3.6 Managing Memory and Non-Linearity

Constructing echo state networks necessarily involves a compromise between two desirable network characteristics: long-term memory of the inputs and ability to model non-linear relationships between the input and target sequences. The simplest method of managing this trade-off is by scaling the weights of the network—particularly the input weights, though the recurrent weights also play a role. Smaller weights force the network dynamics into the more linear region around zero, improving the network's memory, whereas larger weights allow inputs to push the network into the non-linear regions.

While weight scaling is a simple method of dealing with this trade-off, it would be preferable if it were possible to mitigate the need to make such a compromise in the first place. In order to do so, several methods of managing the memory non-linearity trade-off have been explored in the literature.

The first method of augmenting a network in order to manage the trade-off between memory and non-linearity was introduced in the same paper as ESNs themselves [Jae01]. The idea, having the next state be a convex sum of the current state and the update function, was later expanded upon in [Jae+07]. The hidden units in these networks are referred to as *leaky-integrator* units. In a network with leaky-integrator units, the state at time t can be calculated from the input and previous state as

$$\mathbf{x}_t = (1-\alpha)\mathbf{x}_{t-1} + \alpha f(W\mathbf{x}_{t-1} + V\mathbf{u}_t),$$

where $\alpha \in (0, 1]$ is an additional hyper-parameter of the network. Decreasing α has the effect of 'slowing' the dynamics of the network and increases the ability of the network to retain information from long-past inputs [Jae+07; Luk12]. On the other hand, a higher α value means that the network state is more able to adapt the representation in its hidden state to quickly changing inputs. For this reason, it is non-trivial to find the optimal value of α, and it is usually chosen empirically by minimising loss on a validation set.

Another strategy for overcoming the need for this trade-off is composing the update equation to have both linear and non-linear components. The reasoning for this strategy is that since memory is maximised in linear networks, and non-linear function modelling maximised in non-linear ones, a network with both linear and non-linear components offers both these benefits simultaneously, with different aspects of the required computation being delegated to different components of the network. An initial investigation in this direction was carried out in [IY17]: rather than having the activation function applied to each element of the hidden state, the tanh function would be applied to some portion of the hidden units, while the identity function would be applied to the rest. Other implementations of this idea include performing the regression on a concatenation of states from two networks which are identical save for one using a non-linear activation function and the other using the identity and using a convex combination of identity and non-linear activation functions on the hidden states [GGM18].

An alternative approach to solving this problem is to perform the regression on some transformation of the network states, or perform the regression on a concatenation of the network states and such a transformation. One of the simplest techniques using this strategy is to perform regression on a concatenation of the state with the element-wise square of the state [Jae02a]. This introduces non-linearity and allows the approximation of even functions, which is not possible in the standard ESN model without a bias [Dam+12]. An alternative approach to augmenting a network's features was tested in [BVS10], which augments the network with an extreme learning machine based approach: the network state is concatenated with non-linear transformations of random projections of both the state space and the input, giving a richer representation of the data. Moving further from the conventional ESN training paradigm, [BP06] demonstrated that it is possible to improve performance over traditional ESN models by constructing an MLP on top of an echo state network, using the network states as input features for the feedforward network. This detracts from the computational simplicity of the original ESN as training becomes iterative via backpropagation. However, it does also mean that techniques from the domain of feedforward neural networks can be used in order to improve the task-specific performance. An example of this is [RT11b] in which negative correlation learning is used on an ensemble of ESNs in order to improve performance over simple ESN ensembles.

Chapter 4

Memory in Linear Networks

4.1 Introduction

In this chapter, we ask how the memory capacity of a linear echo state network can be inferred from the structure of its weights. In answering this question, we reveal connections between the memory capacity of linear ESNs and the notion of controllability in control theory, and provide an explicit expression for memory capacity of linear systems in terms of properties of its weights W and $\mathbf{v}$.

When constructing models for time series analysis, we are frequently interested in modelling data where there is a large time-lag between the output and some of the inputs required to accurately predict that output. Modelling this kind of long-term dependency was, in fact, one of the motivating factors behind LSTMs, and other gated network structures [HS97]. Unlike these gated networks, the weights of an ESN are not learnt from the data, and the network is unable to learn to distinguish between important and unimportant inputs. It is therefore advantageous to have the network retain as much information as possible, i.e., for the network to have a large memory.

Memory capacity, as discussed in Section 3.3 is one popular way of quantifying the amount of memory in a network. It is well-established that the upper bound for the memory capacity of a linear network is determined by the number of hidden units in the network, and Proposition 3.3.1 gives conditions under which this memory capacity can be achieved. However, it can be difficult to verify whether or not a given network configuration meets the conditions, and the proposition provides no information about the memory capacity in the case where the conditions are not met. Other existing work gives explicit expressions for memory capacity for networks with simple deterministic structures (for instance [RT11a; Sun+12; Liu+18]), but does not provide any information about the more general relationship between a network's weights and its memory capacity.

In this chapter, we expand on [Jae02b] in order to provide a more general method of determining the memory capacity of a system. In doing so, we provide the following contributions:

- We highlight the connections between the memory capacity of linear systems and the notion of controllability in control theory.
- We show that the memory capacity of a linear system is integer-valued[1], and the value is determined by the Jordan form of the recurrent weight matrix W and the direction of $\mathbf{v}$.
- We show that when a network's weights are sampled independently from continuous distributions, the network achieves this maximum memory capacity with probability one.
- For deep linear networks, we provide an expression allowing the construction of an equivalent single-layer network, and by analysing this network, show that deep linear networks whose weights are sampled from continuous distributions also attain the maximum possible memory capacity with probability one.

These contributions show that achieving the maximum possible memory capacity is easy. In fact, almost all (in the measure-theoretic sense) possible weight matrix/vector combinations attain this value. However, this finding is at odds with what is observed in experiments (see, for instance [Jae02b]), and this discrepancy highlights a deeper issue in our understanding of the memory capacity of recurrent networks with regards to the effects of finite precision computation, an issue which is explored further in Chapter 5.

[1]Specifically, we make this claim for memory capacity defined to include the current input u_t as a feature for the regression step.

4.2 Background

In this chapter, we consider ESNs with 1-dimensional input sequences, described by the update equation

$$\mathbf{x}_t = W\mathbf{x}_{t-1} + \mathbf{v}u_t,$$

where $\mathbf{x}_t \in \mathbb{R}^M$ is the hidden state at time $t \in \mathbb{N}$ for some fixed hidden state size $M \in \mathbb{N}$, $u_t \in \mathbb{R}$ is the input at time t, $W \in \mathbb{R}^{M\times M}$ is the recurrent weight matrix and $\mathbf{v} \in \mathbb{R}^M$ is the input weight vector. Since such an ESN is uniquely defined by its weights, we will adopt the practice of referring to the ESN with weights W and $\mathbf{v}$ simply as $(W, \mathbf{v})$. Throughout this chapter, we consider only the case $\rho(W) < 1$, where $\rho(W)$ is the spectral radius of W. We also adopt the notation $\lambda \in S(W)$ to denote that λ is in the set of eigenvalues of the matrix W.

In Proposition 3.3.1, we presented a result from [Jae02b], which states that the memory capacity of $(W, \mathbf{v})$ is M—the maximum possible value—if and only if the matrix

$$\left(\, W\mathbf{v} \, \middle| \, \cdots \, \middle| \, W^M\mathbf{v} \, \right)$$

is full rank. In the rest of this chapter, we will make frequent use of this matrix; we therefore introduce the following definition and notation.

Definition 4.2.1 (Memory Matrix)**.** *Let $(W, \mathbf{v})$ be a linear ESN with M hidden units, we define the memory matrix of $(W, \mathbf{v})$ as*

$$M_{W,\mathbf{v}} \overset{\text{def}}{=} \left(\, W\mathbf{v} \, \middle| \, \cdots \, \middle| \, W^M\mathbf{v} \, \right),$$

i.e., $M_{W,\mathbf{v}}$ is the matrix $M \times M$ matrix whose ith column is the vector $W^i\mathbf{v}$, where W^i is the ith power of the matrix W.

The condition for maximum memory capacity given in Proposition 3.3.1 has a close parallel in control theory, in a result relating to controllability of linear systems. We refer to a discrete-time linear system as controllable if for any state $\mathbf{x} \in \mathbb{R}^M$ there exists a sequence of inputs $u_1, \ldots u_M \in \mathbb{R}$ such that a network driven by those inputs satisfies $\mathbf{x}_M = \mathbf{x}$. A commonly used condition for controllability in linear systems is that the controllability matrix $\left(\, \mathbf{v} \, \middle| \, W\mathbf{v} \, \middle| \, \cdots \, \middle| \, W^{M-1}\mathbf{v} \, \right)$ is of rank M.

For weights $(W, \mathbf{v})$, the memory matrix of a system is simply the controllability matrix left-multiplied by W. As a consequence, assuming W is full rank, the memory matrix is full rank if and only if the controllability matrix is also full rank. The relationship between these matrices allows us to import results from control theory to aid our analysis of the memory capacity of ESNs. In particular, it allows us to make use of a variation of Hautus' Lemma [Hau69], which we state as follows:

Lemma 4.2.1 (Hautus' Lemma, [Hau69], Theorem 1)**.** *Let W be a full rank matrix, then* rank*$(M_{W,\mathbf{v}}) = M$ if and only if for every eigenvalue λ of W,*

$$\operatorname{rank}\left(\left(\begin{array}{c|c} \mathbf{v} & W - \lambda I \end{array}\right)\right) = M.$$

Relative to the standard formulation of the lemma, we have swapped the order of $\mathbf{v}$ and $W - \lambda I$. This change of ordering does not affect the rank of the matrix, but will simplify explanations in later sections. We also state the result in terms of the memory matrix instead of the controllability matrix, which requires that we introduce the restriction to full-rank W.

4.3 Analytically Determining Memory Capacity

In this section, we will show how the memory capacity of an ESN can be inferred from the network structure. As a starting point for our analysis, we show that Proposition 3.3.1 can be generalized to give the memory capacity of a linear system even when the memory matrix $M_{W,\mathbf{v}}$ is not full rank.

Proposition 4.3.1. *Let $(W, \mathbf{v})$ be an ESN with W full rank and $\rho(W) < 1$. The memory capacity of the network is given by*

$$\mathrm{MC}(W, \mathbf{v}) = \operatorname{rank}\left(M_{W,\mathbf{v}}\right).$$

Proof. If $M_{W,\mathbf{v}}$ is full rank, then we have the result by Proposition 3.3.1. Otherwise, say $\operatorname{rank}(M_{W,\mathbf{v}}) = N < M$. We can choose an invertible matrix $P \in \mathbb{R}^{M\times M}$ such that the last $M - N$ rows of $P^{-1}M_{W,\mathbf{v}}$ are zero row-vectors. Define $\widehat{W} \overset{\text{def}}{=} P^{-1}WP$ and $\widehat{\mathbf{v}} \overset{\text{def}}{=} P^{-1}\mathbf{v}$. This gives us for any $k \leq M$, the bottom $M - N$ entries of $\widehat{W}^k\widehat{\mathbf{v}}$ are zero. In fact, this holds for all $k \geq 0$, as for any j either $\widehat{W}^j\widehat{\mathbf{v}} \notin$

$\operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{j-1}\widehat{\mathbf{v}})$ or $\operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{j-1}\widehat{\mathbf{v}}) = \lim_{j' \to \infty} \operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{j'}\widehat{\mathbf{v}})$, and therefore $\lim_{j' \to \infty} \operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{j'}\widehat{\mathbf{v}}) = \operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{M}\widehat{\mathbf{v}})$. For any state $\mathbf{x}_t$ of the network $(W, \mathbf{v})$, we can write

$$\begin{aligned}
\mathbf{x}_t &= \sum_{k=0}^{t-1} W^k \mathbf{v} u_{t-k} \\
&= \sum_{k=0}^{t-1} P \widehat{W}^k \widehat{\mathbf{v}} u_{t-k} \\
&= P \sum_{k=0}^{t-1} \widehat{W}^k \widehat{\mathbf{v}} u_{t-k}.
\end{aligned}$$

This is equivalent to writing the states of the original network as a linear transformation of the states of a network $(\widehat{W}, \widehat{\mathbf{v}})$ driven by the same input sequence. We denote the state of $(\widehat{W}, \widehat{\mathbf{v}})$ at time t by $\widehat{\mathbf{x}}_t$. The two networks have the same approximation power in the sense that for any vector $\mathbf{w} \in \mathbb{R}^M$, we can write $\mathbf{w}^\intercal \mathbf{z}_t = (\mathbf{w}^\intercal P')\widehat{\mathbf{z}}_t$ and $\mathbf{w}^\intercal \widehat{\mathbf{z}}_t = (\mathbf{w}^\intercal P'^{-1})\mathbf{z}_t$, where $\widehat{\mathbf{z}}_t \in \mathbb{R}^{M+1}$ is the input u_t concatenated with the vector $\widehat{\mathbf{x}}_t$, and $P' = \operatorname{diag}(1, P)$.

Though this second network $(\widehat{W}, \widehat{\mathbf{v}})$ also has a hidden state of size M, by construction the last $M - N$ entries of the network state at any time t are just scaled copies of u_t. We can therefore construct a linear echo state network of size N, with the same approximation power as $(\widehat{W}, \widehat{\mathbf{v}})$. To do this, let $\widetilde{W}$ be the top left $N \times N$ sub-matrix of $\widehat{W}$, $\widetilde{\mathbf{v}}$ be the vector consisting of the top N entries of $\widehat{\mathbf{v}}$ and let $\widetilde{\mathbf{z}}_t$ be the input concatenated with that state of $(\widetilde{W}, \widetilde{\mathbf{v}})$ at time t. Since the last $M - N$ entries of $\widehat{\mathbf{x}}_t$ are proportional to u_t and therefore don't contain any additional information, we can construct $\widetilde{\mathbf{w}} \in \mathbb{R}^N$ such that $\mathbf{w}^\intercal \mathbf{z}_t = \mathbf{w}^\intercal P' \widehat{\mathbf{z}}_t = \widetilde{\mathbf{w}}^\intercal \widetilde{\mathbf{z}}_t$, and similarly for any $\widetilde{\mathbf{w}}$ we can construct a corresponding $\mathbf{w}$. This means that the memory capacity of $(W, \mathbf{v})$ is the same as the memory capacity of $(\widetilde{W}, \widetilde{\mathbf{v}})$. It now remains to show that the memory capacity of $(\widetilde{W}, \widetilde{\mathbf{v}})$ is N. By construction, for any $L \geq 1$,

$$\operatorname{rank}\left(\widetilde{W}\widetilde{\mathbf{v}} \,\middle|\, \ldots \,\middle|\, \widetilde{W}^L \widetilde{\mathbf{v}} \right) = \operatorname{rank}\left(\widehat{W}\widehat{\mathbf{v}} \,\middle|\, \ldots \,\middle|\, \widehat{W}^L \widehat{\mathbf{v}} \right). \tag{4.1}$$

Once again we use that for any j, either $\widehat{W}^j\widehat{\mathbf{v}} \notin \operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{j-1}\widehat{\mathbf{v}})$ or $\operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{j-1}\widehat{\mathbf{v}}) = \lim_{j' \to \infty} \operatorname{span}(\widehat{W}\widehat{\mathbf{v}}, \ldots, \widehat{W}^{j'}\widehat{\mathbf{v}})$, this time in conjunction with the fact that $\operatorname{rank}(M_{\widehat{W},\widehat{\mathbf{v}}}) = \operatorname{rank}(P^{-1}M_{\widehat{W},\widehat{\mathbf{v}}}) = \operatorname{rank}(M_{W,\mathbf{v}}) = N$, to conclude that rank$\left(\left(\ \widehat{W}\hat{\mathbf{v}}\ \middle|\ \dots\ \middle|\ \widehat{W}^N\hat{\mathbf{v}}\ \right)\right) = N$. The relationship in Equation 4.1 now immediately yields that rank$(M_{\widetilde{W},\tilde{\mathbf{v}}}) = N$ and it is therefore full-rank. Proposition 3.3.1 can now be applied to get that the memory capacity of $(\widetilde{W}, \tilde{\mathbf{v}})$ is N, and therefore we have established $MC(W, \mathbf{v}) = MC(\widetilde{W}, \tilde{\mathbf{v}}) = N$, completing the proof. □

Using this generalisation of Proposition 3.3.1 allows us to find the memory capacity of a network with any set of weights. In particular, we can derive a relationship between the memory capacity of a linear ESN, the structure of Jordan form of W and the directions of its generalized eigenvectors relative to $\mathbf{v}$.

In what follows, we consider the Jordan matrix decomposition of W, which we write as $W = PJP^{-1}$, where P is the matrix whose columns are the generalized eigenvectors of W, and $J = \operatorname{diag}(J_1, \dots, J_n)$, where each J_i is a Jordan block (we provide supplementary material describing this decomposition in more detail in Appendix A). We also write $P^{-1}\mathbf{v} = \left(\ \mathbf{v}_1^\intercal\ \middle|\ \mathbf{v}_2^\intercal\ \middle|\ \cdots\ \middle|\ \mathbf{v}_n^\intercal\ \right)^\intercal$, where each $\mathbf{v}_i$ corresponds in size to the ith Jordan block. We define the *effective size* of a Jordan block J_i with respect to $\mathbf{v}_i$ as the greatest natural number d_i such that $(\mathbf{v}_i)_{d_i} \neq 0$ (or define it as zero if no such number exists). In what follows, the effective size of a Jordan block J_i is always defined with respect to a particular vector $\mathbf{v}_i$, though when discussing a block's effective size, the reference to $\mathbf{v}_i$ will be omitted for brevity, and we will discuss the property as being the effective size of J_i.

Theorem 4.3.1. *Consider an ESN $(W, \mathbf{v})$, where W is full rank with Jordan matrix decomposition $W = PJP^{-1}$, where $J = \operatorname{diag}(J_1, \dots J_n)$, and each J_i is a Jordan block. Let d_i be the effective size of the Jordan block J_i and define S_λ as the set of J_i with eigenvalue λ. The memory capacity of the network $(W, \mathbf{v})$ is given by the expression*

$$MC(W, \mathbf{v}) = \sum_{\lambda} \max\left(\{d_i : J_i \in S_\lambda\}\right),$$

where the sum is over all distinct eigenvalues of W.

Proof. We consider the rank of $M_{W,\mathbf{v}}$. Since P is full rank, we have

$$\operatorname{rank}(M_{W,\mathbf{v}}) = \operatorname{rank}(P^{-1}M_{W,\mathbf{v}}) = \operatorname{rank}(M_{J,\mathbf{v}'}), \tag{4.2}$$

where $\mathbf{v}' = P^{-1}\mathbf{v}$.

In order to determine the rank of $M_{J,\mathbf{v}'}$, we consider under what conditions rows of the matrix can lose linear independence. We first consider the conditions under which an all-zero row of $M_{J,\mathbf{v}'}$ can exist. Consider a row of $M_{J,\mathbf{v}'}$ which is the ith row in its d-row Jordan block and assume that this row's entries are all zero. Let D be the $d \times M$ sub-matrix of $M_{J,\mathbf{v}'}$ corresponding to this block and write $\mathbf{v}' = (v'_1, \ldots, v'_M)^\intercal$. Each $(D)_{ik}$ satisfies the expression

$$(D)_{ik} = (D)_{(i+1)(k-1)} + \lambda \cdot (D)_{i(k-1)}, \tag{4.3}$$

(see Lemma B.0.1 for the proof of this fact). A consequence of this is that for a row of $M_{J,\mathbf{v}'}$ to be zero, there must be zeros in at least the first $M-1$ entries of the row below. Applying this argument recursively gives that there must be zero entries in at least the first $M-(d-i)$ entries of all rows below i. Since the entries of the bottom row are of the form $(D)_{dj} = \lambda^j v'_d$ for $1 \leq j \leq M$, we have that if any entry of the row is zero (as is the case here), then all entries are zero. By the same reasoning Equation 4.3 gives that if the $(j+1)$th row is all zero, the jth row either is all zero or all non-zero. Since we have already established that each of the rows below the ith are have at least one entry which is equal to zero, they must all be rows of *just* zeros. This can only occur if the corresponding entries of $\mathbf{v}'$ are zero. Therefore, the rank of D is at most the effective size of the Jordan block to which D corresponds.

Furthermore, if W has multiple Jordan blocks with the same eigenvalue, then the span of all rows of $M_{J,\mathbf{v}'}$ corresponding to that eigenvalue is the same as the span of the any of the Jordan blocks for that eigenvalue of largest effective size (proof of this fact is relegated to Lemma B.0.2 in Appendix B). This means that, for each eigenvalue λ, we can choose one of the Jordan blocks for that eigenvalue with largest effective size, and remove all rows corresponding to the other Jordan blocks with corresponding eigenvalue λ while maintaining the same rank as $M_{J,\mathbf{v}'}$. Removing these rows for each λ, as well as removing every all-zero row gives a matrix which we will refer to as $M'_{J,\mathbf{v}'}$. We now have that

$$\operatorname{rank}(M_{J,\mathbf{v}'}) = \operatorname{rank}(M'_{J,\mathbf{v}'}) \leq \sum_{\lambda} \max\left\{d_i : J_i \in S_\lambda\right\}, \tag{4.4}$$

where the inequality is established by counting the rows of $M'_{J,\mathbf{v}}$. We now remove all rows and columns of J for which the correspondingly indexed row of $M_{J,\mathbf{v}'}$ is not in $M'_{J,\mathbf{v}'}$ and call the resulting matrix $\widehat{J}$, similarly remove the corresponding entries of $\mathbf{v}'$ to create the vector $\widehat{\mathbf{v}}$. We can use Lemma 4.2.1 to get that $M_{\widehat{J},\widehat{\mathbf{v}}}$ is full rank by showing that $Y = \left(\widehat{\mathbf{v}} \middle| \; \widehat{J} - \lambda I \right)$ is full rank for any λ which is an eigenvalue of W. For any fixed eigenvalue of W, λ, let j be the index of the last row containing the Jordan block for λ in $\widehat{J}$. By construction, the first entry of this row is non-zero, but all other entries are zero. If we permute the rows of Y so that j becomes the first row, but all other rows maintain their relative positions, and similarly permute the columns of Y so that the $j+1$ column (which is the zero vector) is moved to be the rightmost column, then taking the sub-matrix consisting of all but the last column of this matrix gives a lower triangular matrix with non-zero diagonal. Y is therefore full rank and consequently $\operatorname{rank}(M_{\widehat{J},\widehat{\mathbf{v}}}) = \sum_\lambda \max\{d_i : J_i \in S_\lambda\}$. The memory matrix $M_{\widehat{W},\widehat{\mathbf{v}}}$ is a sub-matrix of $M'_{J,\mathbf{v}'}$, and therefore $\operatorname{rank}(M'_{J,\mathbf{v}'}) \geq \operatorname{rank}(M_{\widehat{W},\widehat{\mathbf{v}}})$. Combining this with Equation 4.2 and Equation 4.4 gives the result:

$$\operatorname{rank}(M_{W,\mathbf{v}}) = \operatorname{rank}(M_{J,\mathbf{v}'}) = \sum_\lambda \max\{d_i : J_i \in S_\lambda\}.$$

$\square$

The above theorem essentially states that there are two ways that the memory capacity of a linear ESN can be diminished: if $\mathbf{v}'$ is configured in such a way that the last elements of a Jordan chain are not excited (i.e., there is a difference between a Jordan block's size and effective size), or if there are multiple Jordan chains for the same eigenvalue. In both cases, this loss of memory capacity is the result of the network not making full use of the state space available to it: in the former case, there is a direction in in $\mathbb{R}^{M+1}$ in which the projection of $\mathbf{z}_t$ is always zero; in the latter, looking at the memory matrix in the basis of generalised eigenvectors reveals directions which are linearly dependent, so once again the network is not using the full state space to store information.

The approach of the proof, using the Jordan matrix decomposition and eliminating linearly dependent rows, is similar to the approach in [PA18] for finding the minimal realisation of linear time-invariant control systems. We consider only the case of full-rank W, however the result holds of rank-deficient W if we change the sum to be over only non-zero eigenvalues of W. This can be seen by constructing a smaller network with the same expressive power as $(W, \mathbf{v})$, as we did in Theorem 4.3.1, and applying the theorem to that network.

A consequence of this theorem is that in order to find the memory capacity of a network where W is full rank and has a simple spectrum, it suffices to count the non-zero entries of $\mathbf{v}^\intercal P$. Once again, we find that this result is similar to a known controllability condition: the Popov-Belevitch-Hautus test—a condition for controllability expressed in terms of eigenvectors and eigenvalues of W and the direction of $\mathbf{v}$ [Hau69]. We state this idea more formally with the following immediate corollary of Theorem 4.3.1.

Corollary 4.3.1. *Consider a linear ESN $(W, \mathbf{v})$, where W is a full rank matrix with no repeated eigenvalues and eigendecomposition $W = P\Lambda P^{-1}$. The memory capacity of the ESN is equal to the number of non-zero entries of $\mathbf{v}^\intercal P$.*

4.3.1 Example: Circulant Reservoirs (SCR and ALR)

Consider the Simple Cycle Reservoir (SCR) as presented in [RT11a] and visualised in Figure 3.1c. For an SCR with M hidden units, W is defined by:

$$W_{ij} = \begin{cases} \gamma, & \text{if } i = j+1 \\ \gamma, & \text{if } i = 1 \text{ and } j = M \\ 0, & \text{otherwise} \end{cases} .$$

For instance, for $M = 5$, the matrix would be written as

$$\begin{pmatrix} 0 & 0 & 0 & 0 & \gamma \\ \gamma & 0 & 0 & 0 & 0 \\ 0 & \gamma & 0 & 0 & 0 \\ 0 & 0 & \gamma & 0 & 0 \\ 0 & 0 & 0 & \gamma & 0 \end{pmatrix} .$$

For any $M \in \mathbb{N}$, the recurrent weight matrix is a circulant matrix, i.e., a matrix where each row is a copy of the row above, but with the entries rotated to the right by one place. A nice property of circulant matrices is that all circulant matrices of size M have the same set of eigenvectors [Bam18]. For $1 \leq k \leq M$, we can write the kth eigenvector of a circulant matrix of size M as

$$\mathbf{p}_k = \frac{1}{\sqrt{M}} \begin{pmatrix} 1 \\ \omega_k \\ \omega_k^2 \\ \vdots \\ \omega_k^{M-1} \end{pmatrix},$$

where $\omega_k \overset{\text{def}}{=} \exp\left(i \cdot \frac{2\cdot\pi\cdot k}{M}\right)$. Note that the matrix whose columns are the eigenvectors of W in the order $k = M, 1, \ldots, M-2, M-1$ is symmetric, since $\omega_k^j = \omega_j^k$ for all $0 \leq k, j \leq M$, and also note that this matrix is exactly the matrix which applies the discrete Fourier transform (DFT) on a signal of length M. There is also a simple expression for the eigenvalues of a circulant matrix: if the first row of W has entries $a_0, a_2, \ldots, a_{M-1}$, then the kth eigenvalue is

$$\lambda_k = a_0 + a_1 \cdot \omega_k + a_2 \cdot \omega_k^2 + \ldots + a_{M-1} \cdot \omega_k^{M-1}, \tag{4.5}$$

which in the case of the SCR gives $\lambda_k = -\gamma \cdot \omega_k$. This gives that the eigenvalues are the M distinct Mth roots of unity, each multiplied by γ, and are therefore all distinct.

For the special case of SCR networks,[RT11a] gives a necessary condition for maximum memory capacity to be achieved (the condition being that $\mathbf{v}$ is not periodic with period less than M). A necessary and sufficient condition—that the square matrix whose columns are rotations of $\mathbf{v}$ is full rank—is also given, but the full circumstances where this condition is met are not explored. Here, we are able to give a different necessary and sufficient condition for $(W, \mathbf{v})$ to have memory capacity M: the DFT of $\mathbf{v}$ has no non-zero components.

As a particular example, this allows us to construct non-periodic $\mathbf{v}$ where the full memory capacity is not achieved. Consider the case where the sum of elements of $\mathbf{v}$ is zero, i.e., $\sum_{i=1}^{M} (\mathbf{v})_i = 0$. In this case, if we consider the first eigenvector (i.e., $k = 0$):

$$\begin{aligned}\mathbf{v}^\intercal \mathbf{p}_k &= \sum_{j=1}^{M} \exp\left(i \cdot \frac{2 \cdot \pi \cdot (j-1) \cdot k}{M}\right) \cdot (\mathbf{v})_j \\ &= \sum_{j=1}^{M} 1 \cdot (\mathbf{v})_j = 0,\end{aligned}$$

and therefore the memory capacity of $(W, \mathbf{v})$ is at most $M - 1$. Note that this is a case that can also be found using the condition in [RT11a]: If the elements of $\mathbf{v}$ sum to zero, then in the matrix of rotations of $\mathbf{v}$, the last row is necessarily equal to the sum of all other rows, and the matrix therefore loses rank. There are, however, other situations where the memory capacity is diminished which our method allows us to see more easily. As an example, consider the case where $M = 4$ and $\mathbf{v} = \begin{pmatrix} 0 & 1 & 1 & 0 \end{pmatrix}^\intercal$, and $\gamma = 0.9$.
The memory matrix for the network $(W, \mathbf{v})$ can be written

$$M_{W,\mathbf{v}} = \begin{pmatrix} 0 & 0.81 & 0.729 & 0 \\ 0 & 0 & 0.729 & 0.6561 \\ 0.9 & 0 & 0 & 0.6561 \\ 0.9 & 0.81 & 0 & 0 \end{pmatrix}$$

By inspection, we can see that the last row of $M_{W,\mathbf{v}}$ is the sum of the first and third rows, minus the second row. Therefore the memory matrix has rank 3, and so $MC = 3$. Though $\mathbf{v}$ is not periodic and its entries don't sum to 0, we note that when $k = 2$, we have

$$\begin{aligned}\mathbf{v}^\intercal \mathbf{p}_k &= \sum_{j=1}^{M} \frac{1}{\sqrt{M}} \exp\left(i \cdot \frac{2 \cdot \pi \cdot k \cdot (j-1)}{M}\right) \cdot (\mathbf{v})_j \\ &= \frac{1}{\sqrt{M}} \sum_{j=1}^{4} \exp\left(i \cdot \frac{2 \cdot \pi \cdot 2 \cdot (j-1)}{4}\right) \cdot (\mathbf{v})_j) \\ &= \frac{1}{\sqrt{4}} \left(\exp\left(i \cdot \frac{2 \cdot \pi \cdot 2 \cdot 0}{4}\right) \cdot 0 + \exp\left(i \cdot \frac{2 \cdot \pi \cdot 2 \cdot 1}{4}\right) \cdot 1\right. \\ &\quad \left. + \exp\left(i \cdot \frac{2 \cdot \pi \cdot 2 \cdot 2}{4}\right) \cdot 1 + \exp\left(i \cdot \frac{2 \cdot \pi \cdot 2 \cdot 3}{4}\right) \cdot 0\right) \\ &= \frac{1}{\sqrt{4}} \left(\exp\left(i \cdot \frac{2 \cdot \pi \cdot 2 \cdot 1}{4}\right) \cdot 1 + \exp\left(i \cdot \frac{2 \cdot \pi \cdot 2 \cdot 2)}{4}\right) \cdot 1\right)\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{\sqrt{4}} \left(\exp\left(i \cdot \pi\right) \cdot 1 + \exp\left(i \cdot 2 \cdot \pi\right) \cdot 1 \right) \\
&= \frac{1}{\sqrt{4}} \left(-1 \cdot 1 + 1 \cdot 1 \right) = 0
\end{aligned}
$$

We already know $MC = 3$, so Corollary 4.3.1 gives that the other values of k must give $\mathbf{v}^\intercal \mathbf{p}_k \neq 0$. Indeed, computing these values, we find $-0.5 + 0.5i$, $-0.5 - 0.5i$ and 1 for $k = 1$, 3 and 4, respectively. We note that though this is an interesting example, it is not a true SCR as defined in [RT11a], as in their specification only $-a$ and a are allowed as entries in the input weight vector for some $a \in \mathbb{R}^+$.

Since the eigenvectors of a circulant matrix are the same regardless of the values of the entries of the matrix, this result extends to any circulant reservoir. In particular, we can apply the same principle to the Adjacent-feedback Loop Reservoir (ALR), albeit reaching a different conclusion than we do for the SCR. We define W in such as reservoir as

$$
W_{ij} = \begin{cases} \gamma, & \text{if } i = j + 1 \\ \gamma, & \text{if } i = j - 1 \\ \gamma, & \text{if } i = 1 \text{ and } j = M \\ \gamma, & \text{if } i = M \text{ and } j = 1 \\ 0, & \text{otherwise.} \end{cases}
$$

For example, for $M = 5$, W would be written as

$$
\begin{pmatrix} 0 & \gamma & 0 & 0 & \gamma \\ \gamma & 0 & \gamma & 0 & 0 \\ 0 & \gamma & 0 & \gamma & 0 \\ 0 & 0 & \gamma & 0 & \gamma \\ \gamma & 0 & 0 & \gamma & 0 \end{pmatrix}.
$$

We need to verify that this matrix has M distinct eigenvalues. By Equation 4.5, we have $\lambda_k = \gamma \cdot \omega_k + \gamma \cdot \omega_k^{M-1}$. Using Euler's formula, along with simple algebraic manipulations, gives

$$
\begin{aligned}
\lambda_k &= \gamma \cdot \left(\omega_k + \omega_k^{M-1} \right) \\
&= \gamma \cdot \left(\exp\left(i \cdot \frac{2 \cdot \pi \cdot k}{M} \right) + \exp\left(i \cdot \frac{2 \cdot \pi \cdot k \cdot (M-1)}{M} \right) \right)
\end{aligned}
$$

$$
\begin{aligned}
&= \gamma \cdot \left(\exp\left(i \cdot \frac{2 \cdot \pi \cdot k}{M} \right) + \exp\left(i \cdot \frac{-2 \cdot \pi \cdot k}{M} \right) \right) \\
&= \gamma \cdot \left(\cos\left(\frac{2 \cdot \pi \cdot k}{M} \right) + i \cdot \sin\left(\frac{2 \cdot \pi \cdot k}{M} \right) + \cos\left(\frac{-2 \cdot \pi \cdot k}{M} \right) + i \cdot \sin\left(\frac{-2 \cdot \pi \cdot k}{M} \right) \right) \\
&= 2 \cdot \gamma \cdot \cos\left(\frac{2 \cdot \pi \cdot k}{M} \right).
\end{aligned}
$$

Since cosine is an even function, we have $\lambda_k = \lambda_{M-k}$ and therefore there are in fact repeated eigenvalues. Furthermore, since the matrix W is symmetric, it is necessarily diagonalisable, and therefore has a simple spectrum. Using Theorem 4.3.1, this means that the memory capacity of the network is equal to the number of distinct eigenvalues of the network (assuming that $\mathbf{v}$ is not orthogonal to any $\mathbf{p}_k$). Taking into account the fact that $\lambda_k = \lambda_{M-k}$ and that $\cos\left(\frac{2 \cdot \pi \cdot M/4}{M}\right) = 0$, the total memory capacity can be evaluated as:

$$
MC = \begin{cases} 2 \cdot n & \text{if } M \text{ is of the form } 4 \cdot n,\ 4 \cdot n - 1 \text{ or } 4 \cdot n - 2 \\ 2 \cdot n + 1 & \text{if } M \text{ is of the form } 4 \cdot n - 3, \end{cases} \tag{4.6}
$$

where in each case n is a natural number.

This differs significantly from the memory capacity given by Lemma 1 in [Liu+12], even accounting for the difference in how we define memory capacity (they use the convention that they do not include u_t as a feature for the regression). We therefore deem it prudent to conduct a numerical exploration of ALR networks in order verify our conclusions.

We run a series of experiments to calculate memory capacity of the networks, i.e., to show how well we are able to extract information about the input history in the worst case scenario where the inputs are a sequence of random values. In order to do so, we use the following setup: we generate a stream of 1000 scalar inputs, generated from a uniform random distribution over the interval $[-1, 1]$. For a variety of different network sizes, we generate an ALR recurrent weight matrix and an input weight vector with entries chosen independently from the set $\{-1, 1\}$, each with probability 0.5. For each reservoir size, we scale the elements to have a desired spectral radius $\rho(W)$, then run the network on the input sequence. Memory capacity is then calculated using the first 750 states and inputs of the network as training data to train the regression weights, then the last 250 are used to calculate the MC from those weights. We repeat this for process for

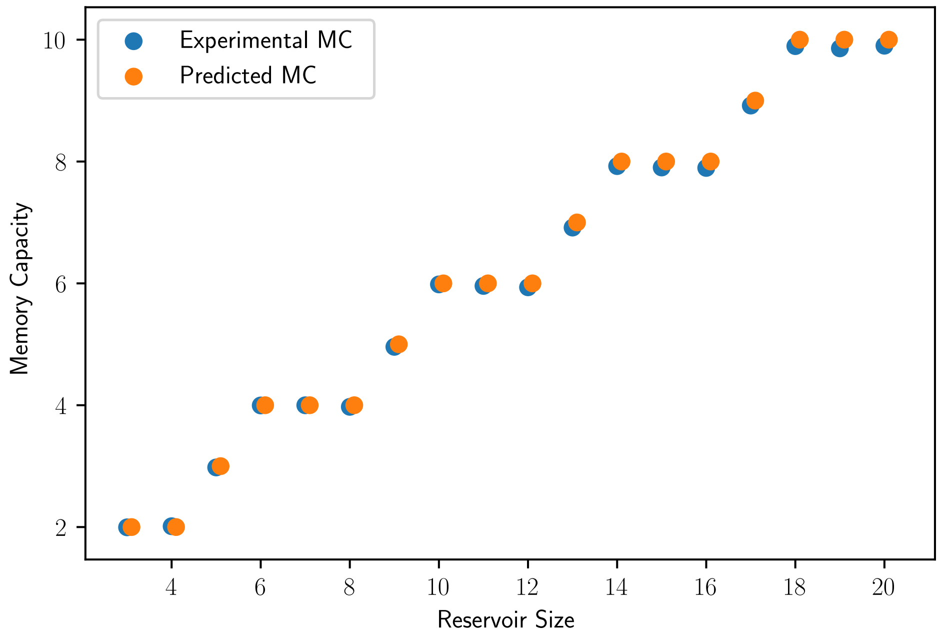


Figure 4.1: Maximum memory capacity of ALR reservoirs found in numerical experiment, and the memory capacity predicted by Equation 4.6. For visual clarity, the predicted values are offset slightly on the horizontal axis.

$\rho(W) \in \{0.2, 0.25, \dots, 0.9, 0.95\}$. For each spectral radius, we conduct 20 runs of the experiment and average the results, we then record the maximum memory capacity achieved for each reservoir size across all spectral radii[2].

The results of this experiment are shown in Figure 4.1 for reservoir size $M = 3, \dots, 20$, along with the the memory capacity values predicted by Equation 4.6. We can see that there is a very close correspondence between the best memory capacity achieved in our experiments and the prediction from our theorem.

The SCR and ALR are the two simplest full-rank deterministic reservoir structures presented in Figure 3.1, it remains an interesting open question as to whether this method can be adapted to the more complex deterministic reservoir structures. This is likely a productive area of future research. In particular, we note that it can be shown that the CRJ and CRHJ recurrent matrices can be

[2]Although in principle the memory capacity should not be sensitive to the spectral radius, in practice finite precision computation means that the effects of varying spectral radius can be substantial. We find in our experiments that for smaller reservoirs, large spectral radii worsen numerical issues, in contrast with larger reservoirs where larger values of $\rho(W)$ are often preferable.

written as block-circulant matrices[3], that is, matrices which can be partitioned into smaller square matrix blocks, and the block-matrix consisting of those smaller blocks is circulant. This allows us to find the eigenvectors of the matrices with relative ease [Tee05], but determining the structure of the eigenvalues is a more difficult proposition which we leave for future research.

4.4 Application to Random Networks

In this section, we show how the insights that we have gained can be used to improve our understanding of the memory capacity of networks under common random initialisation schemes. In particular, we show that for networks where the individual entries of the weights $(W, \mathbf{v})$ are sampled independently from continuous distributions, the maximum possible memory capacity of M is achieved with probability one. We begin with single-layer networks, before going on to show that the same is true of deep networks, even if the same weights are used in each layer.

4.4.1 Shallow Networks

We first consider the case where all entries W and $\mathbf{v}$ are sampled from independent distributions with continuous cumulative distribution functions. In doing so, we will make frequent use of the following lemma.

Lemma 4.4.1. *Let $x_1, \dots, x_N$ be N continuous random variables each with support in $\mathbb{R}$. For any non-constant polynomial $p(x_1, \dots, x_N)$, and for any constant $c \in \mathbb{R}$,*

$$\mathbb{P}\left(p(x_1, \dots, x_N) = c\right) = 0.$$

This lemma is an immediate consequence of the main result in [CT05]. In the proofs that follow, the main value of this lemma is that it yields the fact that if the weights of an ESN are drawn from a distribution with a continuous cumulative distribution function, then useful non-constant polynomials in those entries (such as the determinant) are non-zero with probability one.

Since we make no assumptions about the support of the distributions that we are sampling from, we have no guarantees on the spectral radius of a recurrent

[3]When the jump sizes divide M with no remainder

weight matrix. To combat this, we impose the rule that each recurrent weight matrix has its entries sampled from a some continuous distribution, then the matrix is scaled to have some pre-determined spectral radius between zero and one. Note that this does not affect the directions of the eigenvectors of the matrix, nor whether the spectrum is simple. However, it does mean that we have to be careful when applying Lemma 4.4.1, while it's true for polynomial functions of the entries of the unscaled matrix, it does not necessarily hold for polynomial function of the scaled matrix when the scaling is determined by entries of the matrix.

Theorem 4.4.1. *Let $W_{unscaled}$ be a random matrix and $\mathbf{v}$ a random vector, with entries of both drawn from independent, continuous distributions, and let W be a copy of $W_{unscaled}$ which has been scaled to satisfy $0 < \rho(W) < 1$. With probability one, the ESN $(W, \mathbf{v})$ has memory capacity M.*

Proof. We first argue that W has a simple spectrum (i.e., has no repeated eigenvalues), with probability one. Since the spectrum of W is just the spectrum of W_{unscaled} multiplied by a non-zero scaling factor, W's spectrum is simple if and only if W_{unscaled}'s is. In order for W_{unscaled} to not have a simple spectrum, the characteristic polynomial must have a repeated root, this occurs when the discriminant of the characteristic polynomial is zero. Since the discriminant is also a polynomial in the entries of W_{unscaled}, by Lemma 4.4.1, the probability of choosing entries of W_{unscaled} such that this is the case is zero, and therefore, W is simple with probability one. Using a similar argument with the determinant of the matrix allows us to conclude that W is not just simple but also non-singular with probability one.

Next, assume that W has M distinct non-zero eigenvalues, and let $\mathbf{p}_1, \ldots, \mathbf{p}_M$ be the eigenvectors of W. We require that for each $\mathbf{p}_i$, $\mathbf{p}_i^\intercal \mathbf{v} \neq 0$ with probability one. We get that for the ith eigenvector

$$\mathbf{p}_i^\intercal \mathbf{v} = \sum_{j=1}^{M} (\mathbf{p}_i)_j \cdot (\mathbf{v})_j$$

For any fixed matrix W, we have that each $\mathbf{p}_i$ is fixed, and therefore we have a linear equation in the entries of $\mathbf{v}$. Using Lemma 4.4.1, we have that this expression is non-zero with probability one for each of the M eigenvectors. Since we have the dot product of $\mathbf{v}$ with each of the M eigenvectors is non-zero, the matrix is non-singular and simple—all with probability one— Corollary 4.3.1 gives that the network has maximum memory capacity, also with probability one. □

We can also extend this proof to linear networks with leaky integrator units, as demonstrated in the following corollary.

Corollary 4.4.1. *Let W be a matrix with entries drawn i.i.d. from some continuous distribution then rescaled to ensure that $\rho(W) < 1$, and let $\mathbf{v}$ be a random vector whose entries are also drawn i.i.d. from some continuous distribution. Define a linear network with leaky integrator units by the update equation*

$$\mathbf{x}_t = (1-\alpha)\mathbf{x}_{t-1} + \alpha\left(W\mathbf{x}_{t-1} + \mathbf{v}u_t\right), \tag{4.7}$$

for some $0 < \alpha \leq 1$. The memory capacity of the network is at least $M-1$ with probability one.

Proof. Note that we can re-arrange Equation 4.7 to give

$$\mathbf{x}_t = \left(\alpha W + (1-\alpha)I\right)\mathbf{x}_{t-1} + \alpha\mathbf{v}u_t$$

Let $\mathbf{p}$ be an eigenvalue of W with eigenvalue λ, by definition $(W - \lambda I)\,\mathbf{p} = 0$, and we can therefore reason

$$\begin{aligned} & (W-\lambda I)\mathbf{p} && = 0 \\ \Rightarrow \quad & (\alpha W - \alpha\cdot\lambda I)\mathbf{p} && = 0 \\ \Rightarrow \quad & (\alpha W + (1-\alpha)I - (\alpha\cdot\lambda + (1-\alpha))I)\mathbf{p} && = 0. \end{aligned}$$

Once again using the definition of eigenvalues and their corresponding eigenvectors, $\mathbf{p}$ is an eigenvector of $(\alpha W + (1-\alpha)I)$ with eigenvalue $\alpha\cdot\lambda + (1-\alpha)$. Since the eigenvectors are the same as the non-leaky network, and W has simple spectrum with probability one, Corollary 4.3.1 gives that the memory capacity of the network is M if $\alpha\cdot\lambda + (1-\alpha)$ is non-zero whenever λ is an eigenvalue of W. Since $\alpha\cdot\lambda + (1-\alpha) = 0$ has a unique solution for λ, at most one eigenvalue of the original system is mapped to one, and since W is simple with probability one, the eigenvalue is not repeated and the memory capacity is therefore diminished by at most one, again with probability one. □

We note that the above corollary gives a more general observation about the memory of networks with leaky-integrator units. In general, the memory capacity of a linear leaky-integrator network is the same as that of the corresponding non-leaky network as long as for each eigenvalue λ of W, both λ and $\alpha \cdot \lambda + (1-\alpha)$ are non-zero. We have once again had to take care to deal with consequences of re-scaling the matrix W to a specific spectral radius. If we were to refrain from re-scaling, or place restrictions on our leak rate α, it would be possible to strengthen the result to have maximum memory capacity with probability one.

A common practice when choosing a distribution for ESNs is to have some sparsity constraint on the recurrent weight matrix. For instance, one may choose to initialise the recurrent weight matrix not from a continuous distribution, but from a discrete distribution with a large probability mass at zero. Clearly, in this scenario, it is possible for the criteria for maximal memory capacity to not be met (e.g., if all the recurrent weights are zero). However, results from random matrix theory yield that, if the entries are i.i.d. with zero-mean and bounded fourth moment, the matrix is non-singular [RV08] and has a simple spectrum [Ge17] both with probability $1 - o(1)$ in M. If entries of $\mathbf{v}$ are from a continuous distribution, we can use the same argument as in the above proof to give that the memory capacity is M with high probability. If the entries of $\mathbf{v}$ are also from a discrete distribution, to the best of our knowledge, it remains an open question as to whether maximum memory capacity is achieved with high probability.

4.4.2 Deep Networks

A common strategy used in the construction of neural networks is to compose the network of multiple layers, with each successive layer receiving the previous layer's activations as its inputs. This practice can successfully be applied to recurrent networks, where empirical evidence suggests that composing hidden states in such a manner can lead to network layers whose response to variation in inputs operates on distinct time-scales [HS13] [GM16]. The advantages of depth appear to be present even when the weights of the network are unchanged by training, as in ESNs, and exploring the space of deep ESN architectures has proved to be a productive area of research [MHW17; MSC17; Car+18].

In this section, we consider deep linear networks, with update equations of the form:

$$\begin{aligned}
\mathbf{x}_t^{(1)} &= W_1\mathbf{x}_{t-1}^{(1)} + \mathbf{v}_1\mathbf{u}_t\\
\mathbf{x}_t^{(2)} &= W_2\mathbf{x}_{t-1}^{(2)} + V_2\mathbf{x}_t^{(1)}\\
&\vdots\\
\mathbf{x}_t^{(L)} &= W_L\mathbf{x}_{t-1}^{(L)} + V_L\mathbf{x}_t^{(L-1)},
\end{aligned}$$

where each W_i and V_i is an $M \times M$ matrix, and $\mathbf{v}_1 \in \mathbb{R}^M$. We can re-write this network as a single layer network in the manner shown in Equation 4.8.

$$\begin{pmatrix}\mathbf{x}_t^{(1)}\\ \mathbf{x}_t^{(2)}\\ \mathbf{x}_t^{(3)}\\ \vdots\\ \mathbf{x}_t^{(L)}\end{pmatrix} = \begin{pmatrix} W_1 & 0 & 0 & \cdots & 0\\ V_2W_1 & W_2 & 0 & \cdots & 0\\ V_3V_2W_1 & V_3W_2 & W_3 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \left(\overset{\curvearrowleft}{\prod}_{i=2}^{L} V_i\right) W_1 & \left(\overset{\curvearrowleft}{\prod}_{i=3}^{L} V_i\right) W_2 & \cdots & \cdots & W_L \end{pmatrix} \begin{pmatrix}\mathbf{x}_{t-1}^{(1)}\\ \mathbf{x}_{t-1}^{(2)}\\ \mathbf{x}_{t-1}^{(3)}\\ \vdots\\ \mathbf{x}_{t-1}^{(L)}\end{pmatrix} + \begin{pmatrix}\mathbf{v}_1\\ V_2\mathbf{v}_1\\ V_2V_2\mathbf{v}_1\\ \vdots\\ \left(\overset{\curvearrowleft}{\prod}_{i=2}^{L} V_i\right)\mathbf{v}_1\end{pmatrix} u_t \tag{4.8}$$

Here, $\curvearrowleft$ denotes that in the matrix product, terms with higher indices pre-multiply the lower index terms. We will refer to the $(M \cdot L) \times (M \cdot L)$ recurrent weight matrix in this formulation as $\bar{W}$ and the $(M \cdot L)$ input weight vector as $\bar{\mathbf{v}}$. This form of DeepESN—which we will refer to as a Lin-DeepESN—is a linear version of the ESN explored in a series of papers [GM16; GM17; GMP17]. Since a deep linear network can be expressed as a shallow one of the form in Equation 4.8, we know from [Jae02b] that the upper bound on the memory capacity of the network is $(M \cdot L)$. We devote the rest of this section to showing that this memory capacity is achieved with probability one when the network weights are drawn from continuous distributions.

Independent Weights

Perhaps the most common way of initializing deep networks is to have the weights of each layer sampled independently from some distribution. We consider the case of an L layer network, where each W_i and each V_i and $\mathbf{v}_1$ are independent random variables, with entries of each chosen from a continuous distribution.

Theorem 4.4.2. *Let $(\bar{W}, \bar{\mathbf{v}})$ be a Lin-DeepESN, with L layers, each of size M. If each of the entries of $W_1, \ldots, W_L$, $V_2, \ldots, V_L$ and $\mathbf{v}_1$ are independently* *sampled from continuous distributions, and each W_i is subsequently scaled to satisfy $0 < \rho(W_i) < 1$, then $MC(\bar{W}, \bar{\mathbf{v}}) = M \cdot L$ with probability one*

Proof. We prove the theorem by using Hautus' Lemma to show that $M_{\bar{W},\bar{\mathbf{v}}}$ is full rank. If we had not scaled each W_i, then we would have that $\bar{W}$ has $M \cdot L$ distinct eigenvalues with probability one, since λ is a eigenvalue of W_i if and only if $\det(W_i - \lambda I) = 0$, and for any $\lambda \in \mathbb{R}$, this is a non-constant polynomial in the entries of W_i. However, re-scaling gives us a non-zero probability of the largest eigenvalue being -1 or +1 for multiple W_i. This means we must consider the what happens when an eigenvalue is shared by multiple W_i. For this reason, we split the proof into two cases: the case where eigenvalues are unique to a single W_i, and the case where they are common to multiple W_i.

We begin with the case where $\lambda \in S(W_i)$ for exactly one i. To apply Hautus' lemma we consider the rank of the matrix

$$\left(\bar{\mathbf{v}} \,\middle|\, \bar{W} - \lambda I \right),$$

which using Equation 4.8 we are able to write this $(M \cdot L) \times (M \cdot L + 1)$ as block matrix with blocks of the form

$$\begin{pmatrix} \mathbf{v}_1 & W_1 - \lambda I & 0 & \cdots & 0 \\ V_2\mathbf{v}_1 & V_2 W_1 & W_2 - \lambda I & \cdots & 0 \\ V_3V_2\mathbf{v}_1 & V_3V_2W_1 & V_3W_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \overset{\curvearrowleft}{\prod}_{j=2}^{L} V_j \mathbf{v}_1 & \overset{\curvearrowleft}{\prod}_{j=2}^{L} V_j W_1 & \overset{\curvearrowleft}{\prod}_{j=3}^{L} V_j W_2 & \cdots & W_L - \lambda I \end{pmatrix}. \tag{4.9}$$

For each $2 \leq j \leq L$, for the jth row-block, we subtract V_j copies of the row-block above, preserving the matrix's rank and getting a matrix of the form

$$\begin{pmatrix} \mathbf{v}_1 & W_1 - \lambda I & 0 & \cdots & \cdots & 0 \\ \mathbf{0} & \lambda V_2 & W_2 - \lambda I & 0 & \cdots & 0 \\ \mathbf{0} & 0 & \lambda V_3 & W_3 - \lambda I & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & 0 & \cdots & \cdots & \cdots & W_L - \lambda I \end{pmatrix}. \tag{4.10}$$

Then, for each W_j, we can take the eigendecomposition $W_j = P_j \Lambda_j P_j^{-1}$. If we pre-multiply Matrix 4.10 by $\mathrm{diag}(P_1^{-1}, P_2^{-1}, \ldots, P_L^{-1})$ and post-multiply by diag$(1, P_1, \ldots, P_L)$, we get the matrix

$$\begin{pmatrix} P_1^{-1}\mathbf{v}_1 & \Lambda_1 - \lambda I & 0 & \cdots & \cdots & 0 \\ \mathbf{0} & \lambda P_2^{-1} V_2 P_1 & \Lambda_2 - \lambda I & 0 & \cdots & 0 \\ \mathbf{0} & 0 & \lambda P_3^{-1} V_3 P_2 & \Lambda_3 - \lambda I & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & 0 & \cdots & \cdots & \cdots & \Lambda_L - \lambda I \end{pmatrix}. \tag{4.11}$$

Because by assumption, for $j \neq i$ we have $\lambda \notin S(W_j)$, we get that $\Lambda_j - \lambda I$ is full rank. This, along with rank-preserving row manipulations allow us to zero out blocks on the lower diagonal blocks while preserving the matrix's rank[4]. If $\lambda \in W_i$, we wish to zero out the block to the left of $\Lambda_i - \lambda I$, i.e., the block of the form $\lambda P_i^{-1} V_i P_{i-1}$ (if $i = 1$, this step is not necessary). To do this, we subtract multiples of the rows of $\Lambda_{i-1} - \lambda I$ using the row-blocks above. For instance, in the case where $\lambda \in S(W_2)$, performing the required operations gives a matrix of the form

$$\begin{pmatrix} P_1^{-1}\mathbf{v}_1 & \Lambda_1 - \lambda I & 0 & \cdots & \cdots & 0 \\ C_2 P_1^{-1}\mathbf{v}_1 & 0 & \Lambda_2 - \lambda I & 0 & \cdots & 0 \\ \mathbf{0} & 0 & \lambda P_3^{-1} V_3 P_2 & \Lambda_3 - \lambda I & \cdots & 0 \\ \mathbf{0} & 0 & 0 & \lambda P_4^{-1} V_4 P_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & 0 & 0 & \cdots & \cdots & \Lambda_L - \lambda I \end{pmatrix}. \tag{4.12}$$

Note that or each $i \leq j$, this operation has the effect adding $C_i P_1^{-1}\mathbf{v}_1$ to the first entry in the ith row, and that C_i is full rank with probability one. This gives that the entries of $C_i P_1^{-1}\mathbf{v}_1$ are non-zero with probability one using Lemma 4.4.1, since $(C_i P_1^{-1}\mathbf{v}_1)_i = \sum_{k=1}^{M} (C_i P_1^{-1})_{ik} (\mathbf{v}_1)_k$, a non-constant polynomial in the entries of $\mathbf{v}_1$.

Let k and l be the row and column indices in our matrix of the same form as Matrix 4.12 of the diagonal entry of $\Lambda_i - \lambda I$ which is zero. We can now get the first $M \cdot L$ columns of the matrix into a lower triangular form with non-zero

[4]In particular, we use the fact that if a matrix consists of rows $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_M$, then the row $\mathbf{r}_i$ can be replaced with $\mathbf{r}_i + \sum_{j=1, j\neq i}^{M} \alpha_j \mathbf{r}_j$ without altering the matrix's rank.

diagonal entries by permuting the kth row to the top of the matrix and the lth column to the right, preserving the relative positions of all other rows and columns. A visualisation of these operations is given in Figure 4.2. The diagonal entries of this matrix are the differences between λ and all the other eigenvalues of the W_j's (possibly with repeats), except for the first entry, which we have already established is non-zero with probability one. The matrix is therefore rank $M \cdot L$, and since it was constructed from Matrix 4.9 using only rank-preserving transformations, we therefore have that Matrix 4.9 has rank $M \cdot L$.

Now we consider the case where λ is an eigenvalue of two or more of the recurrent weight matrices. For some $1 \leq i_1 < i_2 < \ldots < i_n, \leq L$, let $\lambda \in S(W_{i_1}) \cap S(W_{i_2}) \cap \ldots S(W_{i_n})$. From Equation 4.11 we can construct via rank-preserving operations the matrix

$$\begin{pmatrix} P_1^{-1}\mathbf{v}_1 & \Lambda_1 - \lambda I & 0 & \cdots & \cdots & 0 \\ C_2P_1^{-1}\mathbf{v}_1 & R_{12} & \Lambda_2 - \lambda I & 0 & \cdots & 0 \\ C_3P_1^{-1}\mathbf{v}_1 & R_{21} & R_{22} & \Lambda_3 - \lambda I & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ C_LP_1^{-1}\mathbf{v}_1 & R_{L1} & R_{L2} & R_{L3} & \cdots & \Lambda_L - \lambda I \end{pmatrix} \tag{4.13}$$

where R_{ij} is all zero if $\lambda \notin S(W_i) \cap S(W_j)$, and otherwise is all zero apart from a single entry $(R_{ij})_{kl}$, where $(\Lambda_i)_{kk} = (\Lambda_j)_{ll} = \lambda$. Each R_{ij} is constructed by first subtracting copies of columns in column blocks where $\Lambda_j - \lambda I$ is in the jth row block and then rows in row blocks where $\Lambda_i - \lambda I$ is in the $i+1$th column block. When $\lambda \in S(W_i) \cap S(W_j)$, these manipulations leave an entry in $(R_{ij})_{kl}$ which is a non-constant polynomial in the entries of V_i and V_j, and is therefore non-zero with probability one.

By the construction Matrix 4.13, we have that the rightmost non-zero entry of each row occurs in a distinct column. Therefore, after we move the all-zero column (i.e., the column which occurs in the column-block containing $\Lambda_{i_n} - \lambda I$, in the column where $(\Lambda_{i_n})_{jj} = \lambda$) to be the rightmost column, we can permute the rows of the matrix to be lower triangular with non-zero diagonal, giving that the matrix is rank $M \cdot L$. A visualisation of the required operations for this proof is shown in Figure 4.3. We therefore have that for all eigenvalues of all W_i the matrix $\begin{pmatrix} \bar{\mathbf{v}} & \bar{W} - \lambda I \end{pmatrix}$ is full rank, and therefore by Hautus' lemma, the memory matrix is of full rank, and the total memory capacity of the network is $M \cdot L$. $\square$

Shared Weights

We now turn our attention to the deep network proposed in [MHW17], in which $V_2 = V_3 = \cdots = V_L$ and $W_1 = W_2 = \cdots = W_L$. In this construction, writing the network in the form Equation 4.8 shows that there are repeated eigenvalues in $\bar{W}$, as the eigenvalues of the lower triangular matrix $\bar{W}$ are the eigenvalues of its diagonal blocks. We find that by dealing with repeated eigenvalues in a manner similar to how we handled them in the previous proof, we are able to show that the network still attains the maximum possible memory capacity with probability one.

Theorem 4.4.3. *Let $(\bar{W}, \bar{\mathbf{v}})$ be a Lin-DeepESN, with L layers, each of size M. Let $W_1 = \cdots = W_L = W$ and $V_2 = \cdots = V_L = V$, with entries of W, V and $\mathbf{v}$ all sampled from continuous distributions with W subsequently scaled to satisfy $0 < \rho(W) < 1$. We have $MC(\bar{W}, \bar{\mathbf{v}}) = M \cdot L$ with probability one.*

Proof. We perform the same rank-preserving manipulations as in the proof for the independent weights case to get a matrix of in the same form as Equation 4.11 and proceed to show that the matrix,

$$\begin{pmatrix} P^{-1}\mathbf{v}_1 & \Lambda - \lambda I & 0 & \cdots & \cdots & 0 \\ \mathbf{0} & \lambda P^{-1}VP & \Lambda - \lambda I & 0 & \cdots & 0 \\ \mathbf{0} & 0 & \lambda P^{-1}VP & \Lambda - \lambda I & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & 0 & 0 & 0 & \cdots & \Lambda - \lambda I \end{pmatrix},$$

has rank $M \cdot L$. For any eigenvalue λ of W, $\Lambda - \lambda I$ is not full rank and and because it appears in all but the first column of the matrix, we are unable to remove the lower diagonal entries using row operations as in the previous proof. However, due to the structure of $\left(\Lambda - \lambda I\right)$ we can we can perform rank-preserving row and column operations to get

$$\begin{pmatrix} P_1^{-1}\mathbf{v}_1 & \Lambda - \lambda I & 0 & \cdots & \cdots & 0 \\ C_2 P_1^{-1}\mathbf{v}_1 & R_{22} & \Lambda - \lambda I & 0 & \cdots & 0 \\ C_3 P_1^{-1}\mathbf{v}_1 & 0 & R_{33} & \Lambda - \lambda I & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ C_L P_1^{-1}\mathbf{v}_1 & 0 & 0 & 0 & \cdots & \Lambda - \lambda I \end{pmatrix} \tag{4.14}$$

where the only non-zero element of each R_{ii} is $(R_{ii})_{kk}$, where k is the index such that $(\Lambda - \lambda I)_{kk} = 0$. In particular, each R_{ii} is constructed by subtracting copies of rows in row block above it (of the form $\Lambda - \lambda I$) and columns in column blocks to the right of it (also of the form $\Lambda - \lambda I$). These manipulations cannot affect the entry R_{kk}, and we know this to be non-zero with probability one since it is $(\lambda P^{-1}VP)_{kk}$ is a polynomial in entries of V.

Inspecting the matrix Equation 4.14, we notice that the rightmost non-zero entry of each row occurs in a distinct column. Therefore, after we move the all-zero column to be the rightmost column, we can permute the rows of the matrix to be lower triangular with non-zero diagonal, giving that the matrix is rank $M \cdot L$. A visualisation of the required operations for this proof is shown in Figure 4.4. $\square$

4.5 Related Work

Since its invention as a measure in [Jae02b], memory capacity has been of great interest to researchers, and numerous empirical studies have been performed to examine the quantity's relationship to other network characteristics [BF14; FBG16; GM16; FG17; GM16; Gal18].

Previous work giving closed-form expressions for memory capacity has been restricted only to networks with simple delay-line structures, or variations thereof [RT11a; Sun+12; Liu+18]. In comparison to the approach in these works, our approach is more general, as we make no assumptions about the network structure other than that the spectral radius of W is less than one. There is another difference between our results and these previous works that is also worth noting: in these previous works, memory capacity was not found to always be integer valued. The reason for this difference, as discussed in Section 4.2, is in the features used in the regression, whereas we use the approach in [Jae02b], using $\mathbf{z}_t$ for our features, these other papers use $\mathbf{x}_t$.

In [WLS04], the memory capacity of orthogonal networks is explored, and a variation of Proposition 3.3.1 is given in terms of the covariance of the hidden states. Whereas we suppose infinite precision, [WLS04] restricts the precision of measurements of the network state by introducing white noise to each $\mathbf{x}_t$. Given the restrictions in precision inherent in real-world computation, this is a desirable property to model, but makes the analysis dependent on a fuller understanding of the structure of W. We also note that the observation that the approximation

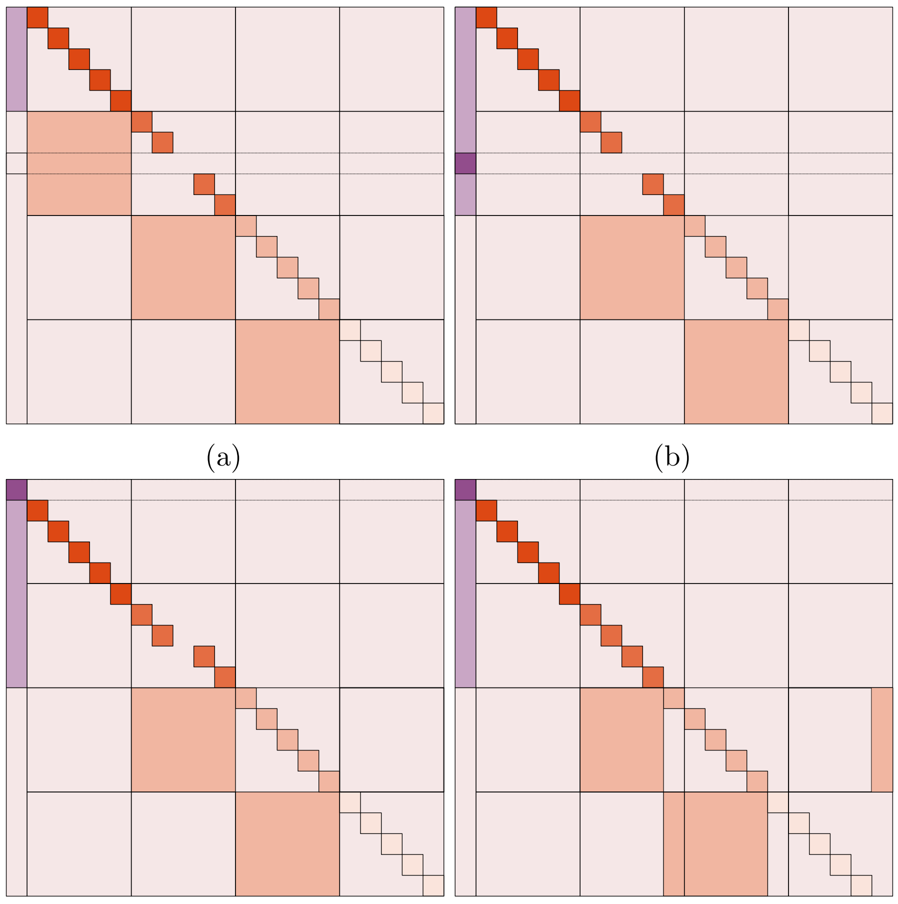


Figure 4.2: Visualisation of the matrix manipulations used in showing that a lin-DeepESN with independently sampled weights has full rank using Hautus' Lemma for a unique eigenvalue. The matrix 4.11 is depicted in a). b) shows the matrix after zeroing out the block to the left of the block containing the eigenvalue under consideration. c) shows the permutation of the rows. d) shows the matrix in c) after permuting the columns, getting the first M columns into lower triangular form.

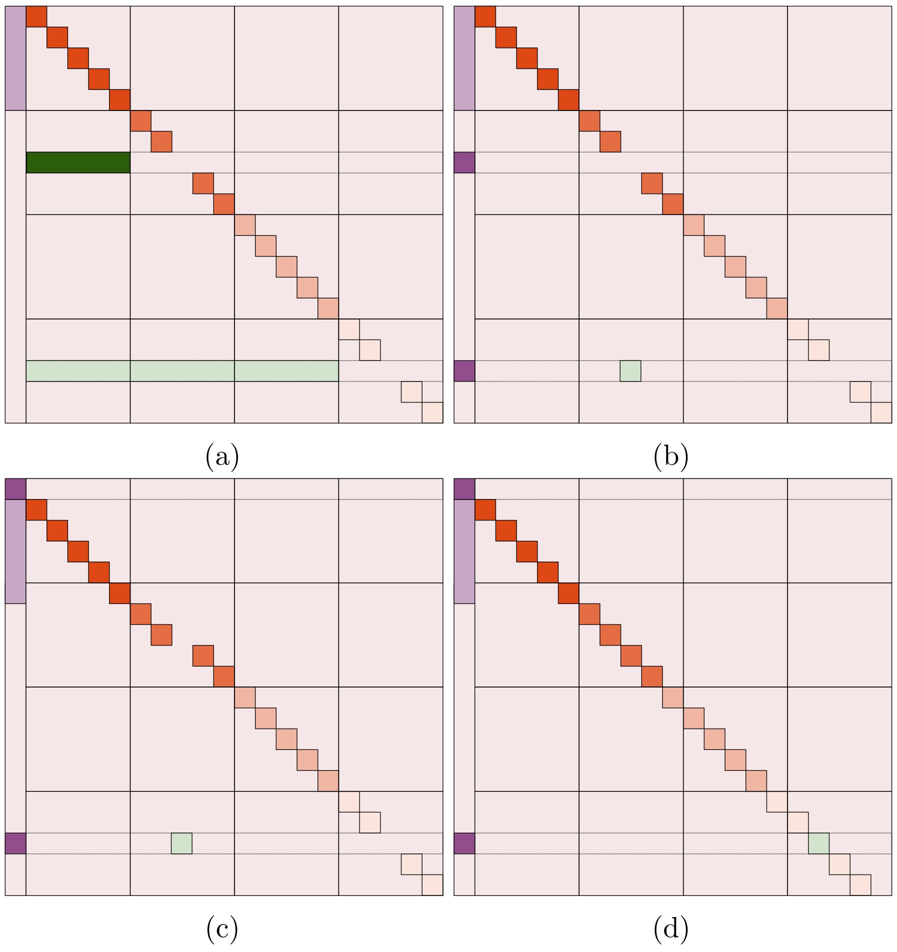


Figure 4.3: Visual representation of the matrix manipulations required to apply Hautus' Lemma when an eigenvalue is shared between multiple W_i (In this case W_2 and W_4. (a) depicts the matrix after the operations zeroing out the rows to the left of each $\Lambda_i - \lambda I$ by subtracting columns. b) shows the matrix after zeroing the rows that were missed by the previous set of operations, this time by subtracting rows from above the current row. c) shows the permutation of the rows d) shows the matrix in c after permuting the columns to get the matrix in lower triangular form.

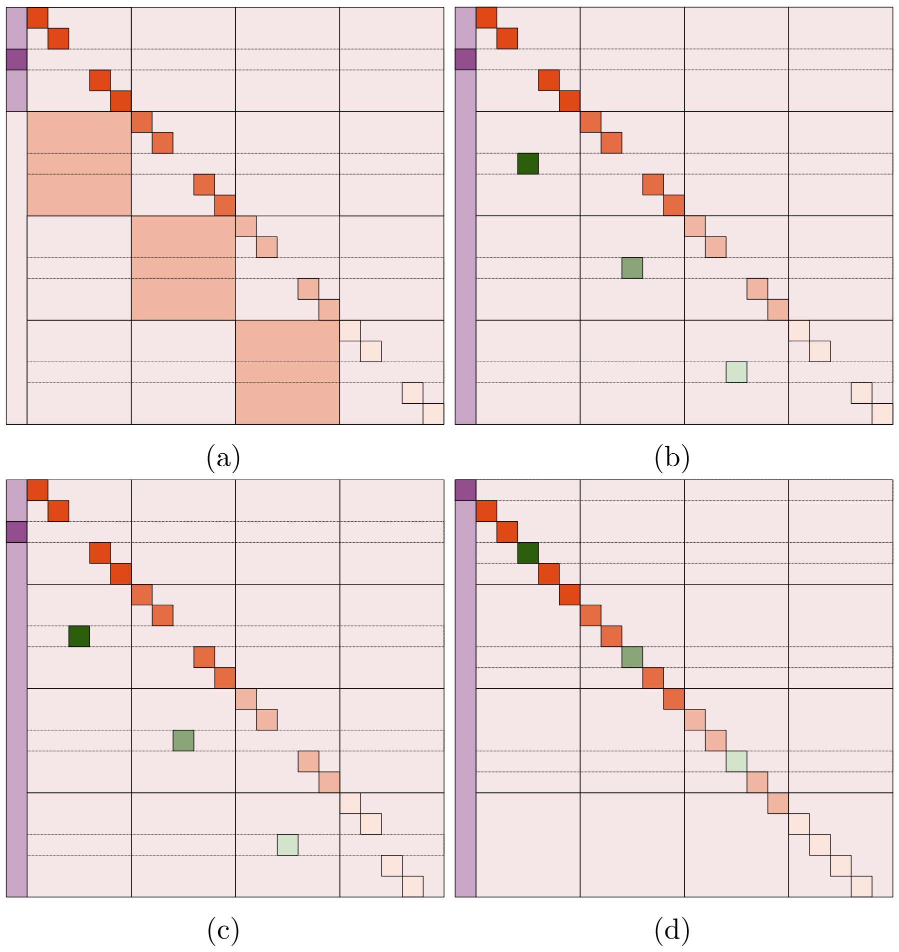


Figure 4.4: Visual representation of the process of showing that matrix with shared weights has full rank using Hautus' Lemma. The matrix 4.14 is depicted in a). b) shows the matrix after zeroing out all but one element of each block below the diagonal by rank-preserving row and column operations. c) shows the permutation of the columns (all occurring in the last block). d) shows the matrix in c after permuting the rows to get the matrix in lower triangular form.

power of a network is determined by its spectrum is found in [ZW08]. Since having the same memory capacity is a consequence of having the same approximation power, Corollary 4.3.1 can be thought of as a special case of their result.

4.6 Conclusion

In this chapter, we explored the relationship between the notion of memory capacity in linear echo state networks and the notion of controllability in control theory. By elucidating the connections between the two concepts, we are able to import existing results from control theory to strengthen our understanding of the relationship between a network's memory capacity and its weights W and $\mathbf{v}$.

We have given examples of how this knowledge can be applied to networks in order to find the theoretical maximum memory capacity of those networks. In doing so, we have shown that networks where the weights are sampled from a continuous random distribution will achieve the maximum possible memory capacity with probability one, for both shallow and deep linear architectures.

It is important to note that in this work we have made the assumption that it is possible to carry out all computation with infinite precision, a luxury not afforded to us by computation in the real world. In practice, the finite precision available to us means that networks fail to achieve their theoretical maximum memory capacity. Due to the exponentially decaying nature of contributions of past inputs, ${}^{MC(W,\mathbf{v})}/_{N} \to 0$ as $N \to \infty$, unless the network is constructed specifically to ensure the existence of entries of $\mathbf{x}_t$ dependent only on inputs from the distant past (such as in a delay-line reservoir). While numerical simulations aid understanding of the restrictions imposed on us by finite precision computation [Jae02b], further work is required to fully understand its effects on memory capacity.

Despite this shortcoming, we have shown that our results are still useful as they give a necessary condition that ESNs must satisfy in order to achieve the best possible memory capacity. In applying our new understanding to SCR and ALR reservoirs, we are able to provide fresh insights into the behaviour of existing ESN designs. While finite precision will remain an issue, the approach examined is this chapter provides future researchers a useful suite of tools for establishing that their model (at least in principle) can attain the maximum possible memory capacity.

Chapter 5

Effects of Depth in Recurrent Neural Networks

5.1 Introduction

It has frequently been observed that deep recurrent neural networks exhibit a phenomenon in which a network's layers respond differently to inputs depending on how deep the layers are in the network. In particular, the layers behave differently with respect to how the influence of an input varies depending upon the delay between the input entering the network and the time-step at which the network state is observed. This class of behaviours is often summarised as the layers of the network 'operating on different time-scales'. Though several different manifestation of this phenomenon have been reported in the literature, its underlying causes are not yet well understood.

In this chapter, we examine the nature of the phenomenon and its relationship to other properties of the network. We focus on two particular manifestations of the phenomenon: sensitivity to input perturbation over time, and memory capacity. In doing so, we attempt to shed light on the causes of the behaviour and the nature of its effects. Though in existing work, the phenomenon has most frequently been observed in non-linear networks, we show that the additional complexity of a saturating non-linearity is not necessary for the phenomenon, and its presence can even diminish the effect.

Our investigation of this phenomenon follows in the spirit of many previous works which empirically examine phenomena related to the behaviour of

ESNs [Boe+11; Cal+13; BF14; FBG16; FG17], including previous work examining Deep Echo State Networks (DeepESNs) [GM16; GMP18; GMS18; GMP19]. Our motivation for this work is threefold: firstly, we aim to shed more light on the root causes of the different times-scales phenomenon. Secondly, by systematically reporting upon its effects, we aim to provide a resource through which the effects can be more thoroughly understood and therefore exploited in future work. Thirdly, by drawing attention to certain aspects of the phenomenon, we hope to inspire future work which more rigorously describes and explains its underlying causes; in particular, by demonstrating that the properties of interest are inherent in the linear version of the networks, we show that future analysis of the phenomenon need not be concerned with the complications which are introduced in non-linear systems.

We outline our contributions in more detail at the start of each section, but at a high level, the contributions of this chapter are:

- We show that in linear networks, though the asymptotic response to perturbations over time of each layer is described by an exponential decay, the short-term behaviour varies substantially between network layers. In deeper network layers, the size of the effect of perturbations initially increases over time, before beginning its exponential decay. We demonstrate that the deeper the layer, the larger the maximum effect of the perturbation becomes, but the longer it takes for this maximum to be reached.

- We expand upon the work of [Gal18] demonstrating the effects of depth on memory capacity. We examine a wider range of hyper-parameter values and show that the phenomenon that they describe also occurs in linear networks. By examining the weights used in reconstructing the inputs and other statistical properties of the states in each layer, we shed light on the causes of differences between the memory capacity curves of different network layers.

- We examine the effect of introducing non-linearity into deep networks. In particular, we highlight the importance of the norms of the feedforward weight matrices in determining the behaviour of the network.

5.2 Background

It was first noted in [HS13] that in deep recurrent networks, layers further from the input layer would recover more slowly from perturbations in the input than those closer to it. This was demonstrated by driving a network with a given input sequence, perturbing the value of the input at a single time-step early in the sequence, then running the network again on the modified input sequence. It was observed that though the difference between the perturbed and unperturbed network states tended to zero in all layers, the convergence was notably slower in deeper layers.

This phenomenon was described as the network layers operating at different *time-scales*. The original work of [HS13] was concerned solely with trained networks with element-wise tanh as their activation function. However, more recent research has shown that this phenomenon, along with several other related phenomena, can also be observed in Deep Echo State Networks (DeepESNs).

In [GM16]—the paper in which the DeepESN was introduced—it was observed that even given the randomly generated weight structure of DeepESNs, the different time-scales phenomenon could be observed. This is an important observation, as it demonstrates that the behaviour seen in [HS13] was not merely a property which emerges through training with backpropagation through time. [GM16] also demonstrated the influence of various network modifications on the different time-scales phenomenon, showing that the effect could be heightened by applying varying strengths of leaky-integration to layers of the network, as well as by applying the unsupervised learning algorithm *intrinsic plasticity* to the network weights. However, an important observation is that though the different time-scales phenomenon is observed in DeepESNs, its effects are different from the typical effect observed in [HS13]. In the standard DeepESN in [GM16], the curves showing the size of the perturbation's effects for different layers have approximately same eventual gradient when plotted on the log scale (albeit offset so that the size of the effect of the perturbation in deeper layers is larger). This behaviour can also be observed in some of the results of [HS13], but it is more common for deeper layers of the trained network to have a slower rate of decay of sensitivity to perturbations. It is plausible that this behaviour is due to some mechanism whereby weights in deeper layers are adapted to be, on average, larger in magnitude, thereby decreasing the contractive effects of a typical application of recurrent weight matrix to the state of that layer. This hypothesis is somewhat supported by the fact that the same behaviour can be observed in deeper network layers in [GM16] when the network is trained with intrinsic plasticity. However, examination of this aspect of the phenomenon is outside the scope of this work.

[Gal18] examined the impact of the depth of a layer in a DeepESN on memory capacity. It was observed that layers deeper in the network had a larger total memory capacity and were able to accurately recall inputs from further in the past than shallower layers. Though this was an important observation, the scope of [Gal18] was limited, reporting the memory capacity curves for a single hyper-parameter configuration, and examining in isolation effects of varying spectral radius on the total memory capacity of network layers.

In [GMP19], another manifestation of the different time-scales phenomenon in DeepESNs was reported. It was observed that in linear DeepESNs, different layers of the network showed different degrees of sensitivity to inputs of different frequencies. In particular, it was demonstrated that the dynamics of deeper layers are 'slower', in the sense that when the network was driven by an input sequence consisting of a superposition of sine waves of the same magnitude but different frequencies, the discrete Fourier transform of hidden units in different layers reveals a greater sensitivity in deeper layers for slower oscillations in the input.

The sensitivity of the DeepESNs to small perturbations has also been examined by numerically evaluating the local Lyapunov exponents of networks with different numbers of layers [GMS18]. Though the analysis of these networks considered only the global behaviour of the networks, rather than isolating individual layers, experimental evidence suggested an increase in sensitivity to changes in initial conditions in deeper layers of the network.

5.2.1 Notation and Network Structure

Similarly to the previous chapter, we define a DeepESN with L layers each with M hidden units each as described in Section 3.5.1 and the update equations are of the form:

$$\begin{aligned}
\mathbf{x}_t^{(1)} &= f(W_1\mathbf{x}_{t-1}^{(1)} + \mathbf{v}_1 u_t) \\
\mathbf{x}_t^{(2)} &= f(W_2\mathbf{x}_{t-1}^{(2)} + V_2\mathbf{x}_t^{(1)}) \\
&\vdots \\
\mathbf{x}_t^{(L)} &= f(W_L\mathbf{x}_{t-1}^{(L)} + V_L\mathbf{x}_t^{(L-1)}),
\end{aligned}$$

where f is the identity function in linear networks, and element-wise tanh in non-linear networks. We refer to $W_1, \dots, W_L \in \mathbb{R}^{M \times M}$ collectively as the recurrent weight matrices of the network, $V_2, \dots V_L \in \mathbb{R}^{M \times M}$ as the feedforward weight matrices and $\mathbf{v}_1 \in \mathbb{R}^M$ as the input weight vector. Throughout this chapter we assume that for $1 \leq i \leq L$, $\rho(W_i) < 1$. Furthermore, we consider networks where $\|V_2\| = \dots = \|V_L\|$, and refer to this value as the feedforward norm. Similarly, we set $\rho(W_1) = \dots = \rho(W_L)$, and refer to this as the recurrent spectral radius. We set $\|\mathbf{v}_1\|$ separately from the norms of the feedforward weight matrices, referring to it as the input weight vector norm. Note that varying this norm by rescaling $\mathbf{v}_1$ is equivalent to rescaling the network's inputs by the same value.

In the linear case, as we saw in the previous chapter, we can construct a single-layer linear network with the same dynamics. For a given Linear Deep Echo State Network (Lin-DeepESN), we refer to this single-layer version as the *flattened* network, and write it as $(\bar{W}, \bar{\mathbf{v}})$, where $\bar{W}$ and $\bar{\mathbf{v}}$ are defined as in Equation 4.8. The state of the flattened network at time t is written as $\bar{\mathbf{x}}_t$. Note that for each $l \in \{1, \dots, L\}$, $\mathbf{x}_t^{(l)} = (\bar{\mathbf{x}}_t)_{(l-1) \cdot M : l \cdot M}$. Here we are using the notation $\mathbf{x}_{a:b}$ to denote, for a vector $\mathbf{x}$ and positive integers a and b, the vector of size $b - a$ comprised of the elements of the vector $\mathbf{x}$ from the element indexed a (exclusive) and the element indexed b (inclusive). This notation for slices of vectors will be used in the rest of this chapter. When it does not cause ambiguity, we may refer to a Lin-DeepESN by its flattened network $(\bar{W}, \mathbf{v})$.

Unless otherwise stated, we use the same network architecture for all experiments in this chapter. We use a 10-layer Lin-DeepESN with 100 hidden units per layer. We initialise each W_i so that each entry is sampled from uniform distribution over an interval centered at zero, then the whole matrix is scaled to satisfy $\rho(W_i) = 0.9$, each V_i is scaled so that $\|V_i\| = 1$ and and $\mathbf{v}_1$ scaled such that $\|\mathbf{v}_1\| = 1$. This exact choice of architecture is somewhat arbitrary, however, we will examine the effects of varying each of these parameters over the course of the chapter.

5.3 Sensitivity to Perturbation

In [HS13], it was observed that deeper layers of a trained non-linear network recovered more slowly from input perturbations than layers closer to the input. In this section, we demonstrate that this phenomenon is not restricted to such networks, and can also be observed in untrained linear networks. In particular, the contributions of this section are as follows:

- We demonstrate the existence of the different time-scales phenomenon in untrained deep linear networks.
- We provide proof that for each layer of such a network, the different time-scales phenomenon is only a short term behaviour, and asymptotically all layers exhibit an exponential decay in their sensitivity to perturbation.
- We provide empirical evidence that in Lin-DeepESNs, the maximal magnitude of the effect a perturbation of a layer's state and the delay at which that magnitude occurs both grow linearly with the distance of the layer from the input. Additionally, we provide evidence that the rate of growth of these two characteristics is dependent upon the spectral radius of the recurrent weight matrices of the network, but is not very sensitive to changes in layer size for even moderately-sized networks.

In order to examine this phenomenon, we consider two identical linear networks driven by input sequences which are also identical for every $t > 1$, and at $t = 1$ have a difference which is of unit magnitude. Denote the states of these two networks at time t as $\mathbf{x}_t$ and $\mathbf{x}'_t$ respectively. The norm of the difference between these two networks can be written as

$$\begin{aligned}\|\mathbf{x}'_t - \mathbf{x}_t\| &= \left\|\sum_{k=0}^{t-1} W^k \mathbf{v} u_{t-k} - \sum_{k=0}^{t-1} W^k \mathbf{v} u'_{t-k}\right\| \\ &= \left\|W^{t-1}\mathbf{v}u_1 - W^{t-1}\mathbf{v}u'_1\right\| \\ &= \left\|W^{t-1}\mathbf{v}(u_1 - u'_1)\right\| \\ &= |u_1 - u'_1| \cdot \left\|W^{t-1}\mathbf{v}\right\| \\ &= \left\|W^{t-1}\mathbf{v}\right\|\end{aligned}$$

Consider a Lin-DeepESN with L layers of M hidden units each and with flattened representation $(\bar{W}, \bar{\mathbf{v}})$, we can write the size of the effect of the perturbation at time t on layer l as

$$\begin{aligned}\left\|(\mathbf{x}_t'^{(l)} - \mathbf{x}_t^{(l)})\right\| &= \left\|(\bar{\mathbf{x}}'_t - \bar{\mathbf{x}}_t)_{(l-1)\cdot M:l\cdot M}\right\| \\ &= \left\|\left(\bar{W}^{t-1}\bar{\mathbf{v}}\right)_{(l-1)\cdot M:l:\cdot M}\right\| .\end{aligned}$$

For future use, we define

$$\mathbf{y}_k^{(l)} \overset{\text{def}}{=} \left(\bar{W}^k \bar{\mathbf{v}}\right)_{(l-1)\cdot M:l\cdot M},$$

and note that

$$\left\|(\mathbf{x}_t'^{(l)} - \mathbf{x}_t^{(l)})\right\| = \left\|\mathbf{y}_{t-1}^{(l)}\right\|. \tag{5.1}$$

Consequently, $\left\|\mathbf{y}_k^{(l)}\right\|$ is the size of the effect of a unit-sized perturbation to the input k time-steps ago on the norm of the state of the lth layer. Not only does this give us a convenient trick for calculating the size of the effect of the perturbation in deeper layers, it also allows us to place bounds on the long-term behaviour of the perturbed network with respect to the original, as described by the following proposition.

Proposition 5.3.1. *Let $(\bar{W}, \bar{\mathbf{v}})$ be an L layer Lin-DeepESN such that for $1 \leq i \leq L$, $\rho(W_i) < 1$. For $1 \leq l \leq L$, there exists $\gamma < 1$, $\alpha > 0$ and t_0 such that for all $t > t_0$*

$$\left\|\mathbf{x}_t'^{(l)} - \mathbf{x}_t^{(l)}\right\| < \alpha \cdot \gamma^t$$

Proof. Let $(\bar{W}_l, \bar{\mathbf{v}}_l)$ be the l layer network composed of the first l layers of $(\bar{W}, \bar{\mathbf{v}})$, and let $\bar{\mathbf{x}}_t'^{(l)}$ and $\bar{\mathbf{x}}_t^{(l)}$ be the perturbed and unperturbed states of $(\bar{W}_l, \bar{\mathbf{v}}_l)$ at time t respectively. We have

$$\begin{aligned}
\left\|\mathbf{x}_t'^{(l)} - \mathbf{x}_t^{(l)}\right\| &= \left\|(\bar{\mathbf{x}}_t' - \bar{\mathbf{x}}_t)_{(l-1)\cdot M:l\cdot M}\right\| \\
&\leq \left\|\bar{W}_l^{t-1} \bar{\mathbf{v}}_l\right\| \\
&\leq \left\|\bar{W}_l^{t-1}\right\| \left\|\bar{\mathbf{v}}_l\right\|.
\end{aligned}$$

Using the fact that the eigenvalues of a block triangular matrix are the eigenvalues of its diagonal blocks (See Appendix A.1 or [Use10] for proof), the construction of the flattened network as given in Equation 4.8 gives that the spectral radius of $\bar{W}_l$ is $\max(\rho(W_1), \ldots, \rho(W_l))$. Hence, by Lemma A.3.1 we have that for all $\gamma > \max(\rho(W_1), \ldots, \rho(W_l))$ there exists t_0 such that for all $t > t_0$, $\left\|\bar{W}_l^{t-1}\right\| < \gamma^{t-1}$. Choosing γ to satisfy $\max(\rho(W_1), \ldots, \rho(W_l)) < \gamma < 1$ and $\alpha = \gamma^{-1} \cdot \|\bar{\mathbf{v}}_l\|$ completes the proof. $\square$

Though the above proposition shows that all layers of a linear network ultimately exhibit an exponential decay in their response to perturbation over time (assuming that the recurrent weight matrices are appropriately scaled), it does not tell us anything about the short-term behaviour of the network. Through numerical experiments, we show that in this shorter time frame we can observe interesting variation between network layers.

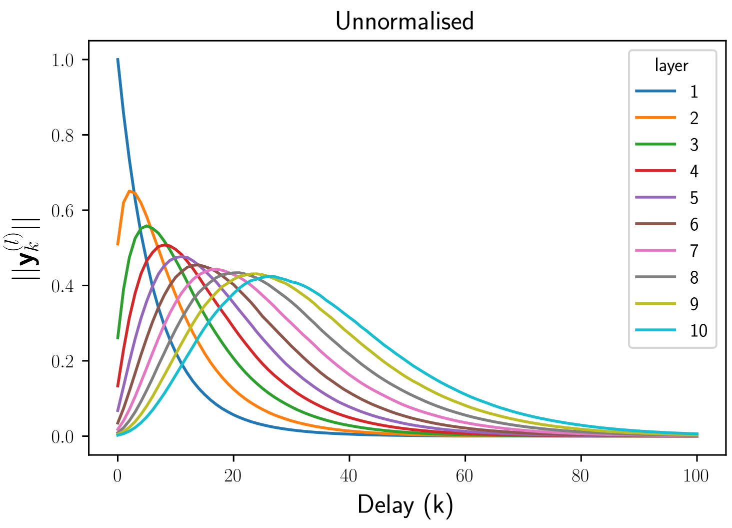
5.3.1 Initial Experiments

For experiments involving sensitivity to perturbation of different layers, we examine the effect of the perturbation of an input on the network by measuring the norm of $\mathbf{y}_k^{(l)}$ for different values of k, essentially measuring how the size of the effect of a perturbation of unit magnitude evolves over time. We conduct this experiment 1000 times, using the network architecture described in Section 5.2.1, resampling the weights at the start of each run and reporting the average over the runs.

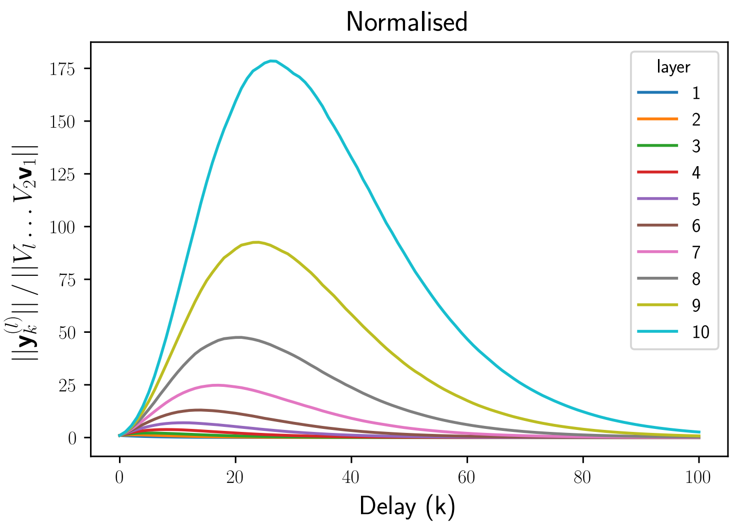
As can be seen in Figure 5.1, a linear untrained network has sufficient complexity to give rise to the different time-scales phenomenon—in the sense that in different network layers the effects of perturbations evolve in different ways, with deeper network layers recovering from perturbations more slowly. In particular, we can see that in the first layer, the sensitivity to perturbation is monotonically decreasing with time. In contrast, higher layers exhibit a phenomenon where the effect of the perturbation on the network state initially increases with time, before beginning the exponential decay dictated by Proposition 5.3.1. Since the scaling of the states in the layers is somewhat arbitrary[1], it is perhaps more insightful to re-normalise by the size of the initial effect of the perturbation (i.e., for the ith layer, divide $\left\|\mathbf{y}_k^{(l)}\right\|$ by $\|V_l V_{l-1} \cdots V_2 \mathbf{v}_1\|$). Doing so produces Figure 5.1b, which shows that this phenomenon can have very significant effects, with the size of the perturbation in the tenth layer growing ≈ 179 times its initial size before beginning its exponential decay.

A natural question to ask is how this phenomenon depends upon the choice of hyperparameters of the network. We can see that scaling the norms of the feed-forward matrices will only have the effect of multiplying the size of perturbations

[1] The scale is arbitrary in the following sense: consider a network with weights $W_1, W_2, \dots W_L$ and $\mathbf{v}_1, V_2, \dots V_l \dots, V_L$ driven by some input, and denote the state of the lth layer at time t as $\mathbf{x}_t^{(l)}$. The state of the network with the same weight matrices, except for V_l, which is replaced with αV_l for some $\alpha \in \mathbb{R}$, the state of the new layer at time t will be $\alpha \mathbf{x}_t^{(l)}$.



(a)



(b)

Figure 5.1: Size of effect of perturbations of input in different layers of a Lin-DeepESN over time. (a) shows the size of the effect for each layer as k varies, (b) shows the same, but each layer is normalised so that the effect of the perturbation at time $t = 0$ has a norm of unit magnitude.

by a constant factor. As such, we can disregard this hyperparameter and consider two others: spectral radius and hidden layer size.

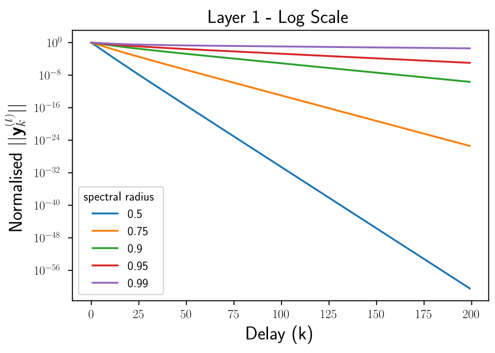
5.3.2 Spectral Radius

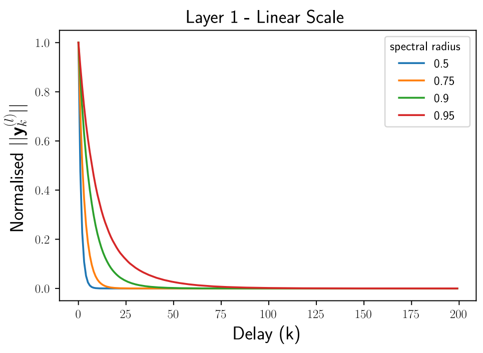
In order to observe the effect of changing the spectral radius, we repeat the previous experiment, but re-scaling the spectral radius for different runs. In particular, we perform runs with spectral radii 0.5, 0.75, 0.9, 0.95 and 0.99. Additionally, we increase the number of layers in our network to 20, allowing us to collect more data about the way that the behaviour of layers changes when they are positioned deeper in the network. As before, for each spectral radius we run 1000 different initialisations of the network and report the average of the results (in the case of the magnitude of the perturbation, we report the average of the log). In Figure 5.2 and Figure 5.3, we visualise the effect of varying the spectral radii of the recurrent weight matrices on the size of the effect of an input perturbation on various network layers. We observe two changes to the network's behaviour as the spectral radius increases: the maximum magnitude of $\left\|\mathbf{y}_k^{(l)}\right\|$ for a given l increases, and time-lag (the value of k) at which this maximum occurs also increases.

To examine this phenomenon more deeply, we plot the the maximum size to which the effect of perturbation grows in each layer against the layer's depth. This is shown in Figure 5.4, where we observe the maximum size attained growing with depth of the network. Interestingly, the maximum height achieved appears to grow linearly with depth of the layer, with the gradient of the line that we observe increasing as the spectral radius is increased towards one.

In order to quantify how linear this relationship is in reality, for each spectral radius we fit a linear regression to to the collected data and compute the squared Pearson's correlation coefficient between the data and and the regression model's prediction for each layer. For each spectral radius, we fit two regressions, one which uses the perturbation sizes for all twenty layers, and a second regression in which we use only layers starting at the first layer for which the proportion of runs where the maximum perturbation occurs at $k = 0$ is less than 0.01. We discuss the motivation for this criterion later in this section.

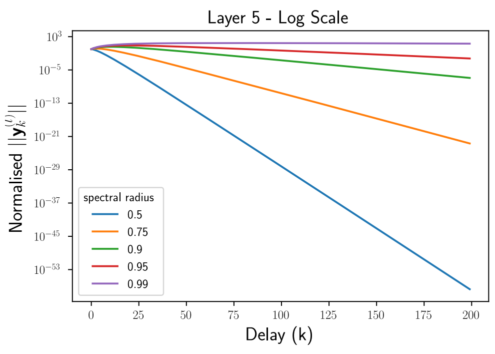
Table 5.1 reports the results of this analysis, and gives strong numerical evidence that there is indeed a linear relationship between the depth of a layer of the network and the expectation of the log of the maximum size of the perturbation

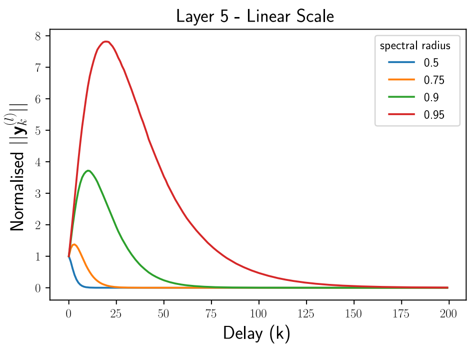




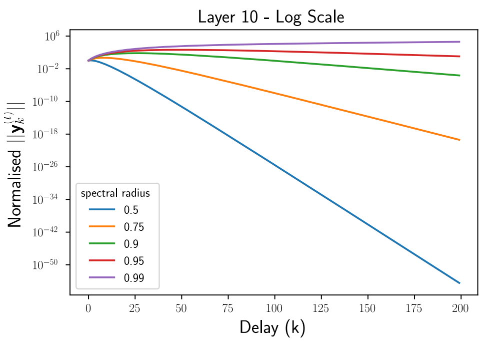
(a)

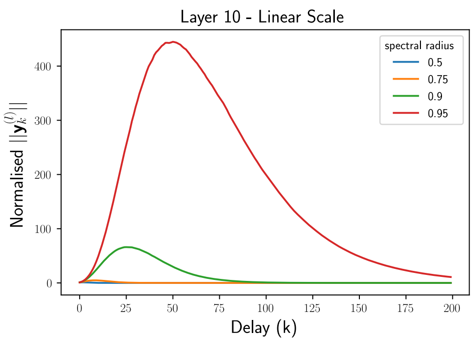
(b)





(c)

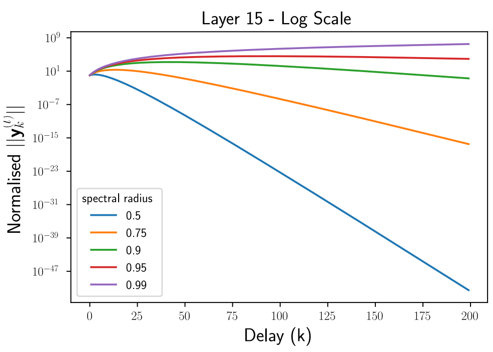




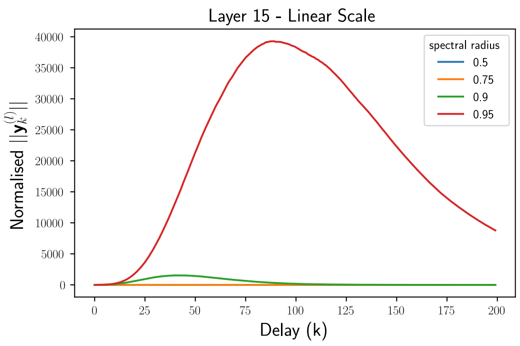
(d)

(e)

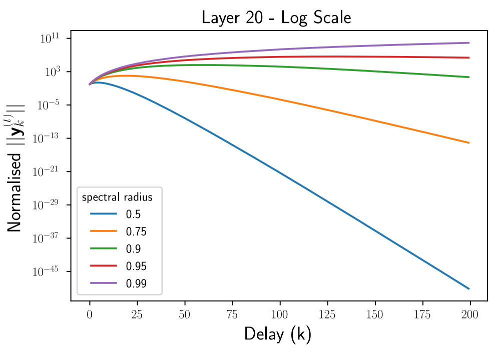
Figure 5.2: Figures showing the effects of spectral radius on a Lin-DeepESN's sensitivity to perturbation of input over different time-delays for the 1st, 5th and 10th layers. On the left, see the behaviour of network layers with spectral radius 0.5, 0.75, 0.9, 0.95 and 0.99, plotted on the log scale, highlighting the eventual exponential decay. On the right, the same data is plotted on a linear scale, in order to more clearly show the growth in perturbation size that occurs in deep layers for shorter delays. For clarity, we omit spectral radius 0.99 on the linear-scale plot.



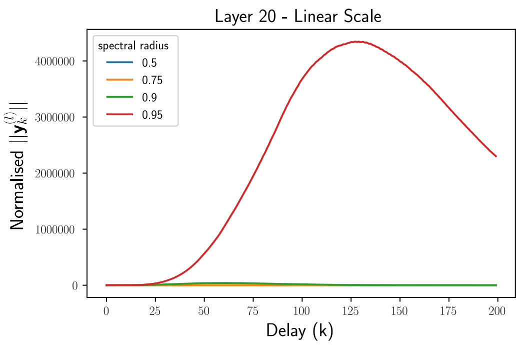
(a)



(b)



(c)



(d)

Figure 5.3: Figures showing the effects of spectral radius on a linear ESN's sensitivity to perturbation of input over different time-delays for the 15th and 20th layers. For further detail, refer to the caption of Figure 5.2.

	All Layers		Using Criterion		
Spectral Radius	Slope	R^2	First Layer	Slope	R^2
0.5	0.099	0.9786	8	0.115	0.9994
0.75	0.327	0.9965	3	0.338	0.9994
0.9	0.650	0.9987	2	0.658	0.9996
0.95	0.872	0.9990	2	0.882	0.9996
0.99	1.269	0.9988	2	1.282	0.9994

Table 5.1: Slope and R^2 of regression on log of size of maximum effect of perturbation

caused by the input.

Next we examine the values of the time-delay k for which this maximum in perturbation size is reached. This is shown in Figure 5.5. Once again, we note that the observed values follows a linear trend (though in some cases with non-linear behaviour in initial layers).

As with the the log-magnitude of the maximum perturbation size, the initial behaviour is non-linear, particularly for spectral radii 0.5 and 0.75. In these cases, the pattern that emerges after the initial layers is suggestive of a linear relationship between the layer number and the delay for which the maximum perturbation occurs, but with the fitted line having a non-zero intercept. Of course, there is a hard floor of $k = 0$ on the y-axis[2]. To be able to disregard this initial behaviour, we may wish to consider only layers for which where the maximum perturbation occurs for k above this value. To do so, we construct the criterion that we include in our regression only layers for which the proportion of runs where the perturbation is at $k = 0$ is less than 0.01. This threshold is somewhat arbitrary, but is preferable to choosing the threshold as zero since it makes the result of using the criterion more robust to changes in the number of runs of the experiment that we conduct. The same reasoning also motivates the use of this criterion in the previous regressions in Table 5.1.

In Table 5.2, we once again observe strong evidence that the relationship is indeed linear. Once again the squared correlation coefficients are highly suggestive of an asymptotically linear relationship, especially if the behaviour of initial layers is discounted.

From the evidence in this section, we conjecture that for a given reservoir size,

[2]In principle, we could consider $\bar{W}^k\mathbf{v}$ for negative values of k, though this is not meaningful in terms of the network's behaviour, and so we refrain from doing so.

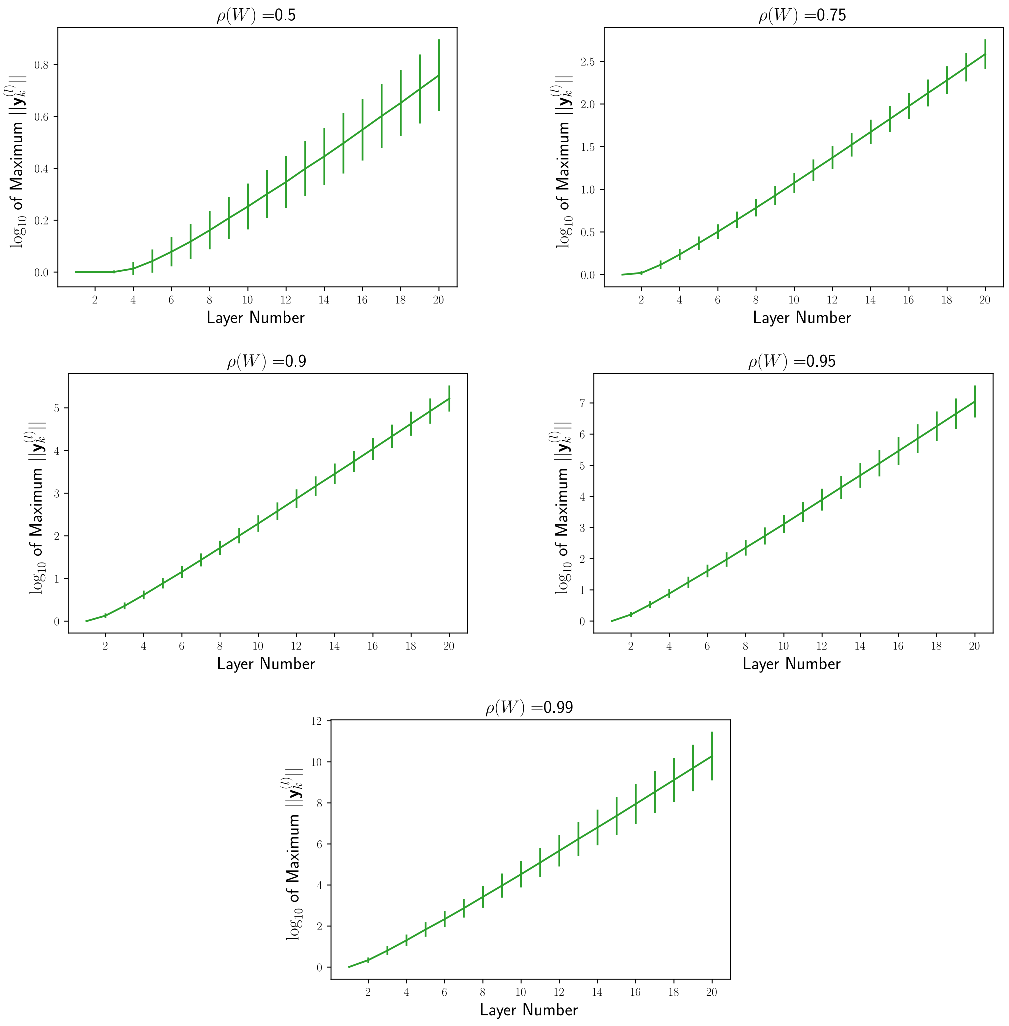
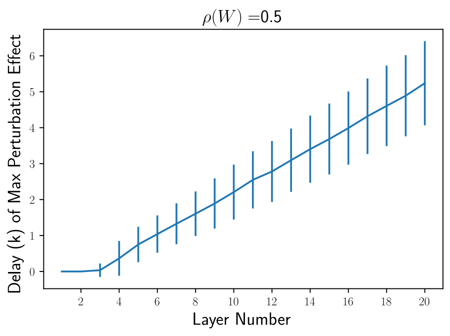
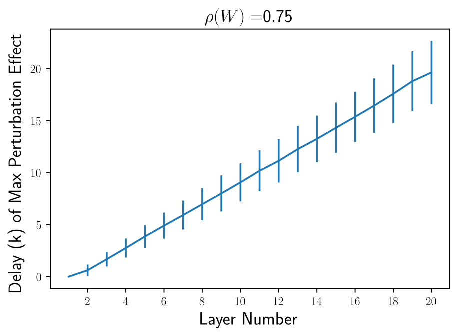


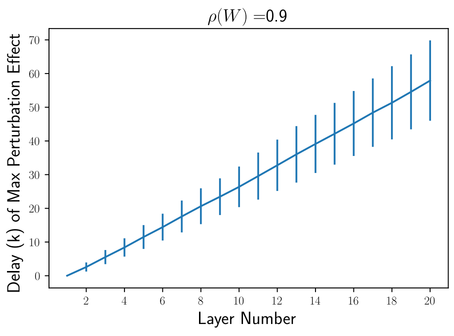
Figure 5.4: Log of size of maximum $\left\|\mathbf{y}^{(l)}{}_k\right\|$ against layer size for 20-layer Lin-DeepESNs with various spectral radii. Logarithms are taken in base 10 in order to allow easier comparison with Figure 5.1



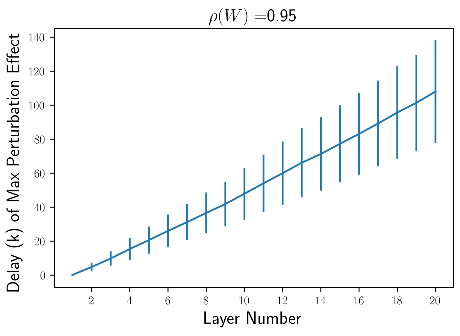
(a) $\rho(W) = 0.5$



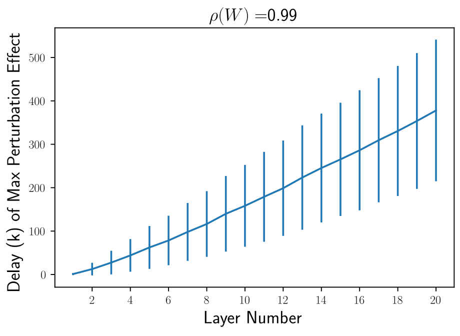
(b) $\rho(W) = 0.75$



(c) $\rho(W) = 0.95$



(d) $\rho(W) = 0.95$



(e) $\rho(W) = 0.99$

Figure 5.5: Value of k at which maximum $\left\|\mathbf{y}_k^{(l)}\right\|$ occurs against layer size for 20-layer Lin-DeepESN.

	All Layers		Using Criterion		
Spectral Radius	Slope	R^2	First Layer	Slope	R^2
0.5	0.291	0.9961	8	0.300	0.9997
0.75	1.052	0.9997	3	1.058	0.9999
0.9	3.060	0.9998	2	3.072	0.9999
0.95	5.696	0.9991	2	5.736	0.9992
0.99	20.203	0.9967	2	20.511	0.9978

Table 5.2: Slope and R^2 of regression on k at which maximum $\|\mathbf{y}\|_k^{(l)}$ is reached

the expected value of $\max_{k\in\mathbb{N}} \log(||\mathbf{y}_k^{(l)}||)$ for a network layer is asymptotically (in l, the depth of the layer, and after normalisation) proportional to the depth of the layer, with some constant of proportionality $C_{\rho(W),M}$, where $C_{\rho(W),M}$ is dependent only on the spectral radius to which the recurrent weight matrices have been scaled and the network size M. We also conjecture a similar relationship between the k at which the maximum perturbation occurs and the layer under consideration, with a different constant which is also determined solely by $\rho(W)$ and M.

Next, we examine the role of the hidden layer size to determine whether it does in fact play a role in determining this relationship.

5.3.3 Hidden Layer Size

In the same way as we did with spectral radius, we look at how the maximum effect of a perturbation, and the delay necessary to reach this maximum are affected by the number of hidden units per layer of the network.

We conduct the same experiments as in the previous section, but this time keeping the spectral radius fixed at 0.9 and varying the layer size, with the number of units per layer taking values between 20 and 200 in increments of 20, repeating the experiment 1000 times for each layer size and reporting the average.

The results of these experiments can be observed in Figure 5.6. In Figure 5.7, we observe the same thing for the value of k at which this maximum perturbation occurs.

Though the network's behaviour in the regards that we are examining appears to be relatively insensitive to the size of the network's layers, the strength of the evidence presented is insufficient for us to conjecture that the maximum perturbation size and the time-delay in its occurrence are truly independent from

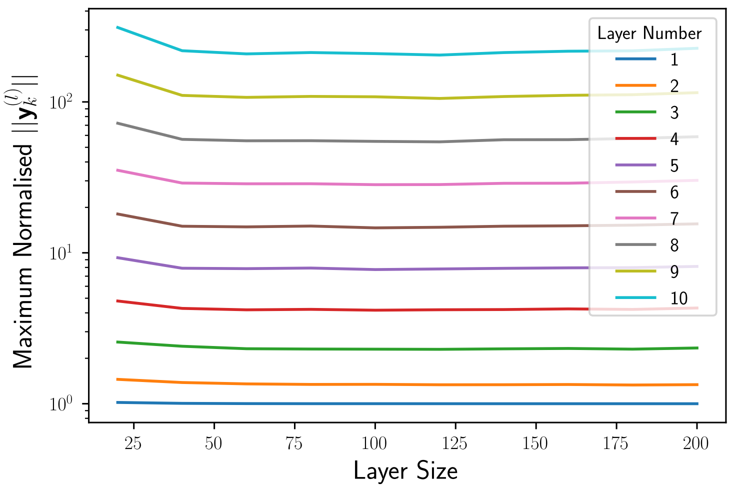


Figure 5.6: Log of maximum perturbation size at each hidden layer when varying number of units per layer.

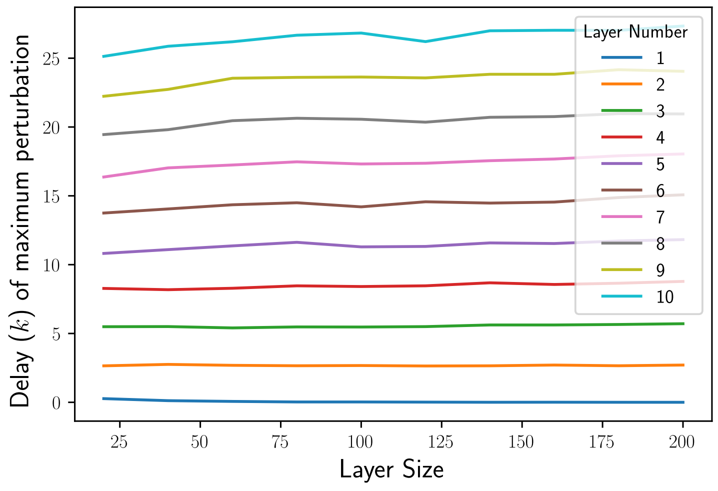


Figure 5.7: Delay until maximum perturbation size is reached for each hidden layer when varying the number of units per layer.

the network size. In particular, we note that Figure 5.6 shows some deviation in behaviour for the smallest of the networks which were examined. It may be the case that the constants of proportionality are of the form $C_{\rho(W),M}$, dependent upon the size of the hidden layers but that for each $\rho(W)$ there exists some constant $C_{\rho(W)}$ such that $\lim_{M\to\infty} C_{\rho(W),M} = C_{\rho(W)}$ and $C_{\rho(W),M}$ approximates this limit even for even modestly sized layers.

5.4 Memory Capacity

The second manifestation of the different time-scales phenomenon which we investigate in this chapter is its effect on the per-layer memory capacity of deep echo state networks. In this section, we build on [Gal18], which showed an increase in memory capacity in deep layers of a DeepESN as compared to the memory capacity of layers closer to the input. The full contributions of this section are listed as follows:

- We show that if a Lin-DeepESN has the maximum possible memory capacity ($M \cdot L$), where M is the number of units per layer and L is the number of layers, then each layer of the network, when considered in isolation, has a memory capacity of M.

- We show that the phenomenon reported in [Gal18] is also exhibited by linear random networks, and the effect is not dependent on the tanh activation function used in their work.

- We examine the different ways in which finite precision computation can cause loss of memory capacity in deep networks, and how these problems can be mitigated against.

We first define the per-layer memory capacity of the network as follows.

Definition 5.4.1 (Layer-wise memory capacity). *Let $(\bar{W}, \bar{\mathbf{v}})$ be an L layer ESN and let $\{u_t\}_{t=-\infty}^{\infty}$ be a sequence of i.i.d. random scalar inputs with finite variance. Define*

$$\mathbf{z}_t^{(l)} \overset{\text{def}}{=} \begin{pmatrix} u_t \\ \mathbf{x}_t^{(l)} \end{pmatrix}$$

and $\mathbf{w}_k^{(l)} \overset{\text{def}}{=} \arg\min_{\hat{\mathbf{w}}_k} \mathbb{E}\left[(\hat{\mathbf{w}}_k^\intercal \mathbf{z}_t^{(l)} - u_{t-k})^2\right]$*, where the expectation is over the distribution of input sequences, then the k-delay memory capacity of the lth layer of the network* $MC_k^{(l)}$ *is defined as*

$$\mathrm{MC}_k^{(l)} \overset{\text{def}}{=} \frac{\mathrm{cov}^2((\mathbf{w}_k^{(l)})^\intercal \mathbf{z}_t^{(l)}, u_{t-k})}{\mathrm{var}(\mathbf{w}_k^\intercal \mathbf{z}_t^{(l)}) \cdot \mathrm{var}(u_t)},$$

The total memory capacity of the lth layer of the network is defined by the infinite sum

$$MC^{(l)} \overset{\text{def}}{=} \sum_{k=1}^{\infty} MC_k^{(l)}$$

We know from the previous chapter that, in theory, a randomly initialised Lin-DeepESN with weights drawn independently from continuous distributions has a total memory capacity of $M \cdot L$ with probability one. Furthermore, we can show that the per-layer memory capacity $MC^{(l)}$ for each layer is bounded by M, and is almost surely M in the case where the weights are randomly initialised from continuous distributions. In order to prove this, we first introduce the following lemma.

Lemma 5.4.1. *Let* $(\bar{W}, \bar{\mathbf{v}})$ *be an L layer neural network with M hidden units per layer. For some* $1 \leq l \leq L$*, let Q be a matrix satisfying* $Q\left(\mathbb{E}\left[\mathbf{z}_t^{(l)}(\mathbf{z}_t^{(l)})^\intercal\right]\right)Q^\intercal = I$*, where* $Q = \Lambda^{-\frac{1}{2}}P^\intercal$ *for an orthonormal matrix P and a full-rank diagonal matrix* Λ*, and let Q satisfy* $\mathbb{E}\left[(Q\mathbf{z}_t^{(l)})_i u_t\right] = \delta_{i1}$*, where* δ_{ij} *is the Kronecker delta. Defining* $\tilde{\mathbf{z}} \overset{\text{def}}{=} Q\mathbf{z}$*, if the input sequence has zero-mean and unit variance, for* $k \geq 1$ *the following equalities hold:*

$$\begin{aligned} MC_k^{(l)} &= \sum_{k=1}^{\infty} \mathbb{E}\left[(y_k^{(l)})^2\right] \\ &= \sum_{i=2}^{M+1} \mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right]^2. \end{aligned}$$

Where $y_k^{(l)}$ *is the optimal linear reconstruction of* u_{t-k} *from the state* $\mathbf{z}_t^{(l)}$*.*

The proof of this lemma is essentially the same as in [Jae02b], so we relegate the details to Appendix C. It is interesting to note that the lemma gives us an alternative way of thinking of the memory capacity of linear networks: as the sum of variances of the optimal input reconstructions (or the variances of the optimal reconstructions divided by the variance of the inputs when they are not normalised to have unit variance). This is something that makes intuitive sense in the extreme cases: when the input can be reconstructed perfectly, y_k will have the same variance as u_{t-k}, and when there is no information that can be extracted about the hidden state, the best strategy is to predict the mean (i.e, zero), meaining the variance will be zero.

With the lemma in hand, we proceed to the main proposition:

Proposition 5.4.1. *Let $(\bar{W}, \bar{\mathbf{v}})$ be a Lin-DeepESN with L layers of M hidden units per layer and memory capacity $M \cdot L$, then*

$$MC^{(l)} = M.$$

Proof. Our proof follows the same structure of the argument made for Proposition 4 in [Jae02b]. To begin, we consider the correlation matrix $R = \mathbb{E}\left[\mathbf{z}_t^{(l)}(\mathbf{z}_t^{(l)})^\intercal\right]$ showing that it has full rank. To do this we first show that $\mathbb{E}\left[\mathbf{x}_t^{(l)}(\mathbf{x}_t^{(l)})^\intercal\right]$ is positive definite (i.e., for all $\mathbf{y} \in \mathbb{R}^M$, $\mathbf{y}^\intercal\mathbb{E}\left[\mathbf{x}_t^{(l)}(\mathbf{x}_t^{(l)})^\intercal\right]\mathbf{y} > 0$). Since the memory capacity of $(\bar{W}, \bar{\mathbf{v}})$ is $M \cdot L$ by assumption, the memory matrix of the network (as defined in Definition 4.2.1) must be full rank. This in turn implies that for any layer l, the sub-matrix $\left(M_{\bar{W},\bar{\mathbf{v}}}\right)_{(l-1)\cdot M:l\cdot M}$ is full row-rank. For each $i \in \mathbb{N}$, define

$$\mathbf{r}_i \overset{\text{def}}{=} \left(\bar{W}^i\bar{\mathbf{v}}\right)_{(l-1)\cdot M:l\cdot M}$$

Now take an arbitrary non-zero vector $\mathbf{y} \in \mathbb{R}^M$. Due to $\left(M_{\bar{W},\bar{\mathbf{v}}}\right)_{l\cdot M:(l+1)\cdot M}$ being full rank, for at least one $i \in \{1, \ldots, M \cdot L\}$, we must have $\mathbf{y}^\intercal\mathbf{r}_i \neq 0$. If we consider the state of the network at time t, when the network has been driven by a left-infinite input sequence from time $-\infty$, the independence of u_i and u_j when $i \neq j$ yields

$$\begin{aligned}
\mathbf{y}^\intercal\mathbb{E}\left[\mathbf{x}_t^{(l)}(\mathbf{x}_t^{(l)})^\intercal\right]\mathbf{y} &= \mathbf{y}^\intercal\mathbb{E}\left[\left(\sum_{k=0}^{\infty}\mathbf{r}_k u_{t-k}\right)\left(\sum_{k=0}^{\infty}\mathbf{r}_k u_{t-k}\right)^\intercal\right]\mathbf{y} \\
&= \mathbf{y}^\intercal\mathbb{E}\left[\sum_{k=0}^{\infty}\left(\mathbf{r}_k u_{t-k}\right)\left(\mathbf{r}_k u_{t-k}\right)^\intercal\right]\mathbf{y} \\
&= \mathbb{E}\left[\mathbf{y}^\intercal\sum_{k=0}^{\infty}\left(\mathbf{r}_k u_{t-k}\right)\left(\mathbf{r}_k u_{t-k}\right)^\intercal\mathbf{y}\right]
\end{aligned}$$

$$\begin{aligned}
&= \mathbb{E}\left[\left(\mathbf{y}^\intercal \sum_{k=0}^{\infty} (\mathbf{r}_k u_{t-k})\right)^2\right] \\
&= \mathbb{E}\left[\left(\sum_{k=0}^{\infty} \mathbf{y}^\intercal (\mathbf{r}_k u_{t-k})\right)^2\right] \\
&= \mathbb{E}\left[\sum_{k=0}^{\infty} (\mathbf{y}^\intercal (\mathbf{r}_k u_{t-k}))^2\right] \\
&= \sum_{k=0}^{\infty} \mathbb{E}\left[(\mathbf{y}^\intercal (\mathbf{r}_k u_{t-k}))^2\right] \\
&= \sum_{k=0}^{\infty} (\mathbf{y}^\intercal \mathbf{r}_k)^2 \cdot \mathbb{E}\left[u_{t-k}^2\right] \\
&> 0.
\end{aligned}$$

Therefore $\mathbb{E}\left[\mathbf{x}_t^{(l)}(\mathbf{x}_t^{(l)})^\intercal\right]$ is positive definite and must be full rank. Note that $\mathbb{E}\left[\mathbf{x}_t^{(l)}(\mathbf{x}_t^{(l)})^\intercal\right]$ is the matrix making up the bottom-right $M \times M$ sub-matrix of R. Combining this with the fact that the top row is all-zero, except for the first entry, we get that R is full-rank.

Since R is a full-rank symmetric matrix, it can be decomposed $R = P\Lambda P^\intercal$ where Λ is a diagonal matrix and P is a matrix whose columns are form an orthonormal basis of $\mathbb{R}^M$ (and therefore $P^\intercal = P^{-1}$). We define $Q = \Lambda^{-\frac{1}{2}}P^\intercal$, noting that $QRQ^{-1} = I$ and also define $\tilde{\mathbf{z}}_t^{(l)} = Q\mathbf{z}_t^{(l)}$. Due to the structure of the top row of R, it is also possible to choose P in order to impose the restriction on Q that $(Q^{-1}\mathbf{z}_t^{(l)})_1 = (\mathbf{z}_t^{(l)})_1$, and therefore the conditions for Lemma 5.4.1 hold.

Define $\bar{\mathbf{x}}_t^{(l)}$ as the $((l-1)\cdot M)$-length vector consisting of the states of the first $l-1$ layers of $(\bar{W}, \bar{\mathbf{v}})$, and define $\bar{V}_l$ as the $M \times ((l-1)\cdot M)$ matrix whose last M columns are the columns of V_l and all other entries are zero. Finally, if we define $\bar{W}_l$ and $\bar{\mathbf{v}}$ to be the matrix and vector such that $(\bar{W}_l, \bar{\mathbf{v}}_l)$ is the network with $((l-1)\cdot M)$ hidden units whose states are the first $((l-1)\cdot M)$ states of $(\bar{W}, \bar{\mathbf{v}})$ under the same input, we are able to write

$$\begin{aligned}
\mathbf{x}_t^{(l)} &= W_l \mathbf{x}_t^{(l)} + \bar{V}_l \bar{\mathbf{x}}_t^{(l-1)} \\
&= \sum_{k=0}^{\infty} W_l^k \bar{V}_l \bar{\mathbf{x}}_{t-k}^{(l-1)}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{k=0}^{\infty} W_l^k \bar{V}_l \sum_{j=0}^{\infty} \bar{W}_l^j \bar{\mathbf{v}}_l u_{t-(j+k)} \\
&= \sum_{k=0}^{\infty} \sum_{j=0}^{\infty} W_l^k \bar{V}_l \bar{W}_l^j \bar{\mathbf{v}}_l u_{t-(j+k)} \\
&= \sum_{k=0}^{\infty} \left(\sum_{m=0}^{k} W_m^k \bar{V}_l \bar{W}_l^{m-k} \bar{\mathbf{v}}_l \right) u_{t-k}. \qquad (5.2)
\end{aligned}
$$

Equation 5.2 decomposes the state of the lth layer into the sum of contributions from individual inputs, in the same way as $\mathbf{x}_t = \sum_{k=0}^{\infty} W^k \mathbf{v} u_{t-k}$ does in the single layer case. This allows us to derive an expression for components of $\tilde{\mathbf{z}}_t^{(l)}$ in terms of the contributions of individual inputs. For $i \neq 1$, we have

$$
\begin{aligned}
(\tilde{\mathbf{z}}_t^{(l)})_i &= Q_{i1} u_t + \sum_{j=2}^{M+1} Q_{ij} (\mathbf{x}_t^{(l)})_{j-1} \\
&= Q_{i1} u_t + \sum_{j=1}^{M} Q_{i(j+1)} \left(\sum_{k=0}^{\infty} \sum_{m=0}^{k} W_l^m \bar{V}_l \bar{W}_l^{m-k} \bar{\mathbf{v}}_l u_{t-k} \right)_j \\
&= Q_{i1} u_t + \sum_{k=0}^{\infty} \sum_{j=1}^{M} Q_{i(j+1)} \left(\sum_{m=0}^{k} W_l^m \bar{V}_l \bar{W}_l^{m-k} \bar{\mathbf{v}}_l \right)_j u_{t-k} \\
&= \sum_{k=1}^{\infty} \sum_{j=1}^{M} Q_{i(j+1)} \left(\sum_{m=0}^{k} W_l^m \bar{V}_l \bar{W}_l^{m-k} \bar{\mathbf{v}}_l \right)_j u_{t-k}, \qquad (5.3)
\end{aligned}
$$

Define $\beta_{ik} = \sum_{j=1}^{M} Q_{i(j+1)} \left(\sum_{m=0}^{k} W_l^m \bar{V}_l \bar{W}_l^{m-k} \bar{\mathbf{v}}_l \right)_j$. With this definition, we have for $2 \leq i \leq M+1$

$$
(\tilde{\mathbf{z}}_t^{(l)})_i = \sum_{k=0}^{\infty} \beta_{ik} u_{t-k},
$$

and we can use Equation 5.3 to get. Since memory capacity of a linear network is invariant to changes in scale of the input, we can assume without loss of generality that the input has unit variance, giving

$$\begin{aligned}
\sum_{k=1}^{\infty} \beta_{ik}^2 &= \sum_{k=1}^{\infty} \beta_{ik}^2 \cdot \underbrace{\mathbb{E}\left[u_{t-k}^2\right]}_{=1} \\
&= \sum_{k=1}^{\infty} \sum \mathbb{E}\left[(\beta_{ik} \cdot u_{t-k})^2\right] \\
&= \mathbb{E}\left[\sum_{k=1}^{\infty} \left(\sum_{j=1}^{M} Q_{i(j+1)} \left(\sum_{m=0}^{k} W_l^m \bar{V}_l \bar{W}_l^{m-k} \bar{\mathbf{v}}_l \right)_j u_{t-k} \right)^2 \right] \\
&= \mathbb{E}\left[\left(\sum_{k=1}^{\infty} \sum_{j=1}^{M} Q_{i(j+1)} \left(\sum_{m=0}^{k} W_l^m \bar{V}_l \bar{W}_l^{m-k} \bar{\mathbf{v}}_l \right)_j u_{t-k} \right)^2 \right] \\
&= \mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i^2 \right] \\
&= 1
\end{aligned}$$

where fourth line is derived from the third by noting that the cross terms in the square are all zero because $\mathbb{E}\left[u_{t-k}u_{t-l}\right] = \mathbb{E}\left[u_{t-k}\right]\mathbb{E}\left[u_{t-l}\right] = 0$ whenever $l \neq k$. Putting this all together, we have

$$\begin{aligned}
MC^{(l)} &= \sum_{k=1}^{\infty} \sum_{i=2}^{M+1} \mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k} \right]^2 \\
&= \sum_{k=1}^{\infty} \sum_{i=2}^{M+1} \mathbb{E}\left[\left(\sum_{j=0}^{\infty} \beta_{ij} \cdot u_{t-j} \cdot u_{t-k} \right) \right]^2 \\
&= \sum_{k=1}^{\infty} \sum_{i=2}^{M+1} \mathbb{E}\left[(\beta_{ik} \cdot u_{t-k})^2\right] \\
&= \sum_{i=2}^{M+1} \sum_{k=1}^{\infty} \beta_{ik}^2 \cdot \mathbb{E}\left[u_{t-k}^2\right] \\
&= \sum_{i=2}^{M+1} \sum_{k=1}^{\infty} \beta_{ik}^2 \\
&= M
\end{aligned}$$

$\square$

Though we do not prove it here, it seems likely that this result can be extended to a more general case, where the memory capacity of entries of a hidden state

	Spectral Radius				
Layer Number	0.5	0.75	0.9	0.95	0.99
1	39.70	62.17	75.67	79.38	81.13
4	49.85	91.94	99.76	99.58	97.62
7	55.21	99.00	99.75	99.49	96.47
10	59.50	99.86	99.75	99.38	94.89

Table 5.3: $MC^{(l)}$ for layers of a deep linear ESN with 100 units per layer for various spectral radii.

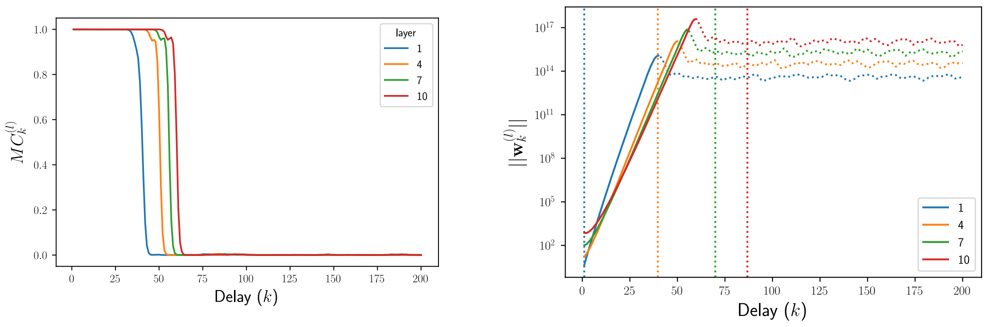
(in this case, the M entries corresponding to a single layer in an $M \cdot L$ flattened network) can be inferred from the row-rank of the corresponding rows in the memory matrix of the network.

The above proposition can be used in conjunction with results from Chapter 4. For instance, combining it with Theorem 4.4.2 gives that when the weights of a Lin-DeepESN are sampled from a continuous distribution, the network layers will each have memory capacity M with probability one.

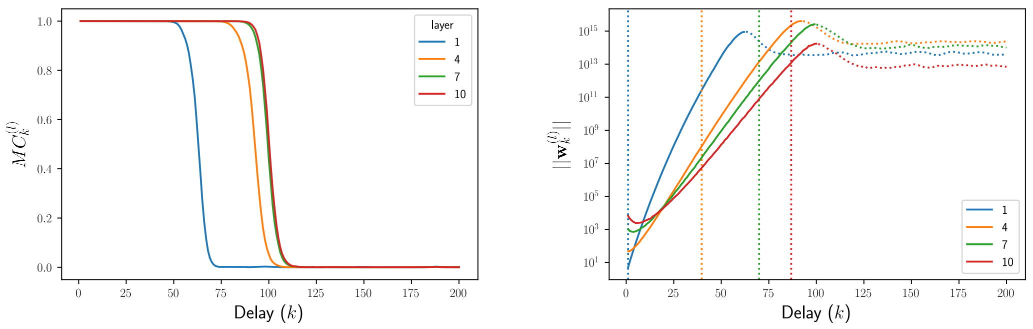
Note that the proposition does not consider the finite precision available to us, nor does it give us any information about the shape of memory capacity curve. It is, however, useful in the following sense: it guarantees that, given infinite precision computation and that the memory capacity of the flattened network is $M \cdot L$, each of the layers have the same memory capacity, and therefore any differences between the total memory capacity of different layers *must* be a result of finite precision computation. Knowledge of this fact can be used to inform the approach that is used when empirically evaluating the memory capacity of a network.

We begin our investigation by plotting the memory capacity curve of networks using the standard network architecture described in Section 5.2.1. This is presented in Figures 5.8 and 5.9, where we show the k-delay memory capacities of layers of a Lin-DeepESNs averaged over 1000 runs with different initialisations of the network. Each network has 10 layers of 100 hidden units each, with weights randomly initialised for each run. Each run is conducted by driving the network with a sequence of 6000 inputs generated from a uniform distribution over the interval $[-1, 1]$. We subject this sequence to a 5000:1000 train/test split, and discard the first 100 states of the training set in order to eliminate initial transient behaviour in the network.

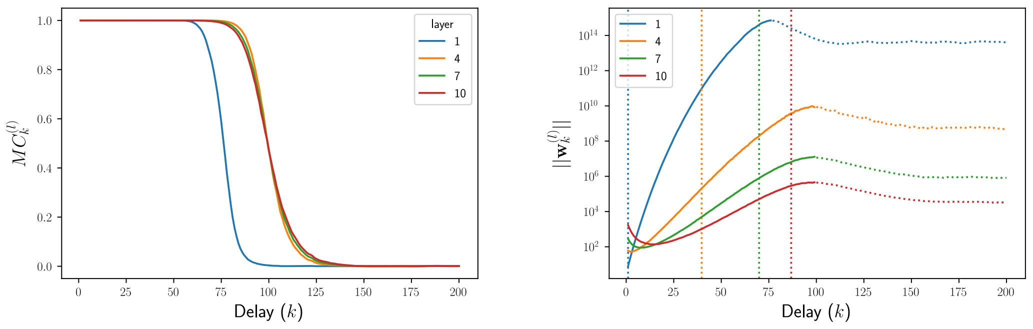
We remark upon two notable aspects of the network behaviour which can be



(a) Spectral Radius = 0.5

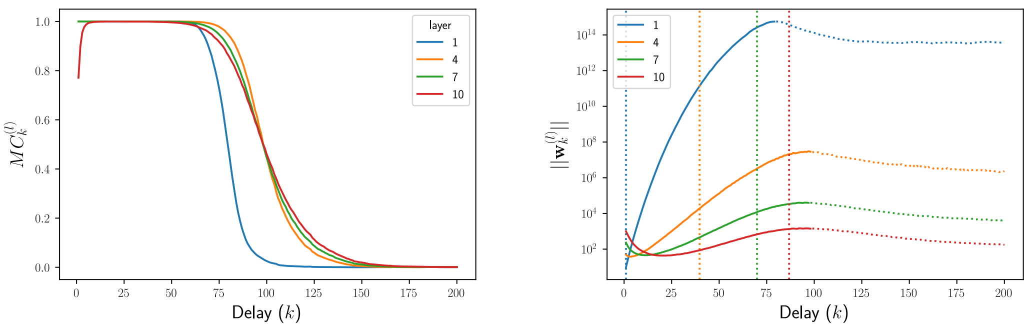


(b) Spectral Radius = 0.75

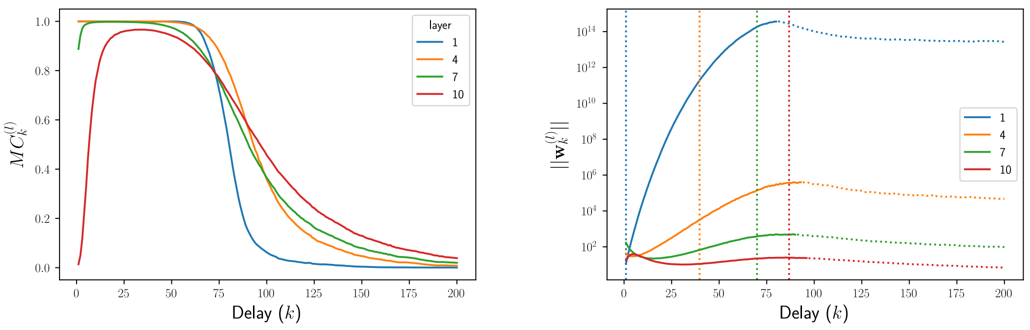


(c) Spectral Radius = 0.9

Figure 5.8: Layer-wise memory memory capacities of layers of a 10-layer Lin-DeepESN with 100 units per layer for various spectral radii. The left images show the values of $MC_k^{(l)}$ for $l = 1, 4, 7, 10$ while varying k. The right images show the norm of $\mathbf{w}_k^{(l)}$ for each k for those same networks. The line showing the size of the norm becomes dotted when $MC_k^{(l)}$ falls below 0.5. The vertical lines show the value of k for which an input at time $t - k$ would cause the maximum perturbation for the layer with the corresponding colour.



(a) Spectral Radius = 0.95



(b) Spectral Radius =0.99

Figure 5.9: Per layer memory capacities of layers of a 10-layer network with 100 units per layer for various spectral radii. Further detail can be found in the caption for Figure 5.8

observed in Figure 5.8 and Figure 5.9:

1. **Effects of Depth and Spectral Radius on Total Memory Capacity:** Like [Gal18] found in the non-linear case, we find that deeper layers of the network have longer memories than those closer to the input. In particular, Table 5.3 shows that for all spectral radii, layers 4, 7 and 10 have greater layer-wise memory capacity than the first layer. However, we do not find this to be a universal phenomenon, i.e., greater depth and higher spectral radius do not guarantee a higher total memory capacity. In fact, for spectral radii 0.9, 0.95 and 0.99, the maximum memory capacity amongst the observed layers occurs in the fourth layer of the network, with spectral radius 0.99 seeing a marked degradation in total memory capacity in deeper layers.

2. **Effects of Depth and Spectral Radius on Size of Reconstruction Weights:** Examining the norms of the weights required for the reconstructions, we find that for deeper layers, the growth is not monotonic with respect to k. From the previous section examining the size of perturbations, this is something that might have been predicted: the weight norms shrink initially as the size of the contribution of u_{t-k} grows. However, we also observe from the figures that it is not the case that the delay for which the perturbation is greatest is also the delay for which the weights are smallest, with the k for which the perturbation is maximised being larger than the k for which the reconstruction weights have the largest norm. To understand why this may be the case, we consider what is happening to $\bar{W}^k\bar{\mathbf{v}}$ as $k \to \infty$. Not only is the size of the contribution geting smaller, i.e., $\left\|\bar{W}^k\bar{\mathbf{v}}\right\| \to 0$, the directions in which these vectors point is getting less varied, i.e., there exists a low-dimensional subspace[3] of $\mathbb{R}^{M \cdot L}$ such that the component of $\bar{W}^k\mathbf{v}/\left\|W^k\mathbf{v}\right\|$ orthogonal to that subspace decays exponentially (See Appendix C.3 for additional detail).

The first of these two observations is of particular interest to us, as understanding what causes the layer-wise memory capacity to deviate from the value

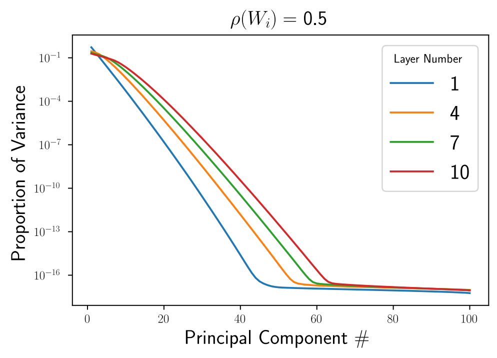
[3]The dimension of the subspace is determined by the spectrum of W (assuming that $\mathbf{v}$ has a non-zero component in the directions of all generalised eigenvectors of W). For W where the entries are sampled from a continuous distribution, the dimension of the subspace is almost surely one or two, depending on whether there is a single eigenvalue of largest magnitude or two forming a complex conjugate pair. Though once again rescaling adds complications to the picture, and allows the dimension of the subspace to grow linearly with depth for deeper layers.

predicted by Proposition 5.4.1 is an important first step in designing strategies to mitigate against the effect. As such, we further investigate the causes of the phenomenon through additional experiments.

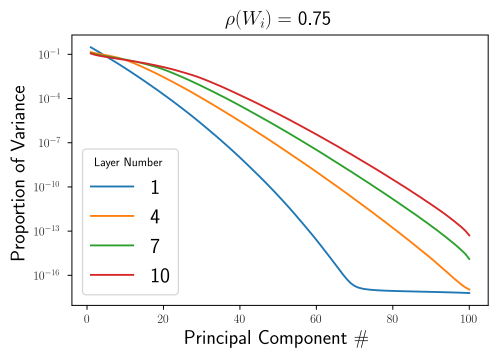
We first address why deeper network layers are able to remember inputs further in the past than the first layer of the network. In order to do so, we perform Principal Component Analysis (PCA) on the collected network states for each layer. For a matrix $X \in \mathbb{R}^{N \times M}$ with $M < N$, PCA constructs an orthonormal basis for the space $\mathbb{R}^M$, with the basis vectors ranked in order of the amount of variance the rows of X have in the direction of those vectors. If $X \in \mathbb{R}^{N \times M}$ is a matrix whose rows are states of a layer of an DeepESN when driven by a given input sequence, PCA first finds the direction in $\mathbb{R}^M$ in which the empirical variance in those states is maximised—we refer to this as the first principal component. For $2 \leq i \leq M$, the ith principal component is defined iteratively as the direction orthogonal to the first $(i-1)$ principal components for which variance is maximised. The variance in each of these directions $(\sigma_1, \dots \sigma_M)$ are the *singular values* of X.

Using our standard 10 layer, 100 unit-per-layer architecture, we run 1000 randomly initialised networks on the same input sequence of length 6000 whose entries are draw independently from a uniform distribution over $[-1, 1]$. The states of each layer are collected at each time step, the first 100 time-steps are discarded and for each layer of interest, PCA is performed on the 5900×100 matrix with rows consisting of the transposed states of that layer. We consider the proportion of variance (PoV) attributed to each principal component, i.e., for $1 \leq i \leq M$, we are interested in the value $\sigma_i / \sum_{j=1}^{M} \sigma_j$.

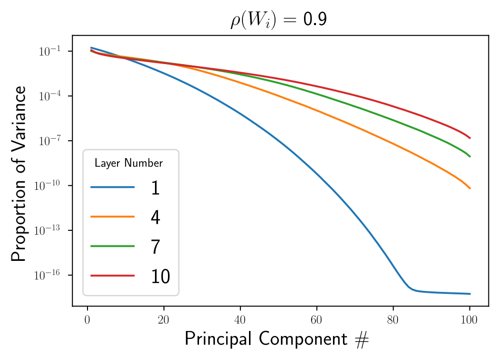
The mean proportion of variance attributed to each principal component over all runs is presented in Figure 5.10 for networks with spectral radii 0.5, 0.75, 0.9, 0.95 and 0.99. For all observed spectral radii, in the first layer the proportion of variance that principal components are responsible can be observed dropping faster than exponentially with the index of the principal component, until eventually levelling off. It is worth noting that the highest proportion of variance is of the order 10^{-1}, the levelling off occurs at values on the order 10^{-17} and—since we are conducting experiments using NumPy's 64-bit arrays with a 52 bit mantissa[Oli06]—computation is performed with approximately 16 significant digits in the decimal representation. This is highly suggestive of the idea that there are directions where the variance is attributable to rounding error, and therefore the whole space is not being utilised.



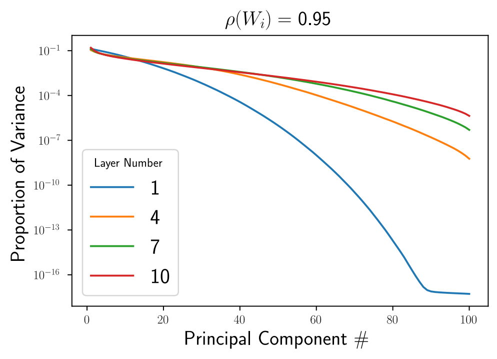
(a)



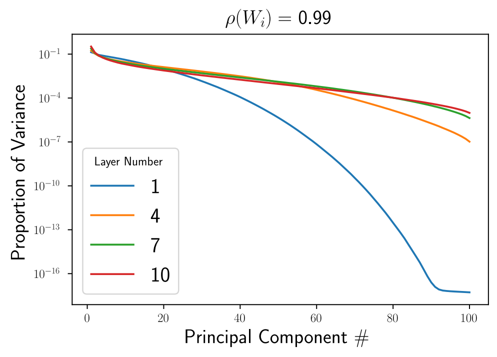
(b)



(c)



(d)



(e)

Figure 5.10: Proportion of average variance that each principal component is responsible for in layers of an DeepESN.

This problem puts a restrictive upper limit on the memory capacity of ESN layers, as demonstrated in Figure 5.11, and it appears that the rate of decay of PoV amongst the principal components varies little with layer size. Successive doublings the size of the network layers offers diminishing returns, both on improvement in memory capacity and on decrease in decay rate of PoV, approaching some limit.

We also observe that the drop-off in PoV gets slower for each successive network layer. This means that PCA cannot explain the reduced memory capacity seen in the deep layers of the network with $\rho(W_i) = 0.99$, as observed in Figure 5.9b. Indeed, in Figure 5.10e we observe that PoV for the 10th layer of that network has has slowest decay amongst all observed spectral radius/layer pairings. Though we are unable to offer a complete explanation for the cause of this phenomenon, we conjecture that it can be thought of as being the opposite extreme from the issue that we find with low spectral radii. That is to say, it appears to be related to signals not dying out quickly enough, and the state space becoming in a sense 'cluttered'. In support of this idea, we note that:

- Unlike the issue encountered when the spectral radius is low, we are able to mitigate this problem by increasing the size of layers of the network. This can be observed in Figure 5.12, where we observe that for spectral radius 0.99, the observed degradation in performance disappears for networks with layers of size 200 and 400 and a drop in performance occurs in the 50 unit per layer network for spectral radius 0.95.

- The same phenomenon can be observed, albeit to a lesser extent, in single layer networks with high spectral radius (and in the first layer of Lin-DeepESNs). An example of this is shown in Figure 5.13. In order to create this effect in a single-layer network, the spectral radius had to be chosen to be very close to one, and the network size had to be relatively small, whereas the effect is significantly more pronounced in deeper layers of a deep network.

5.5 Effects of Non-Linearity

Throughout this chapter thus far, we have consider only the case where the activation function is the identity function. This allows us to gain valuable insights into the root causes of observed phenomena, but it is not representative

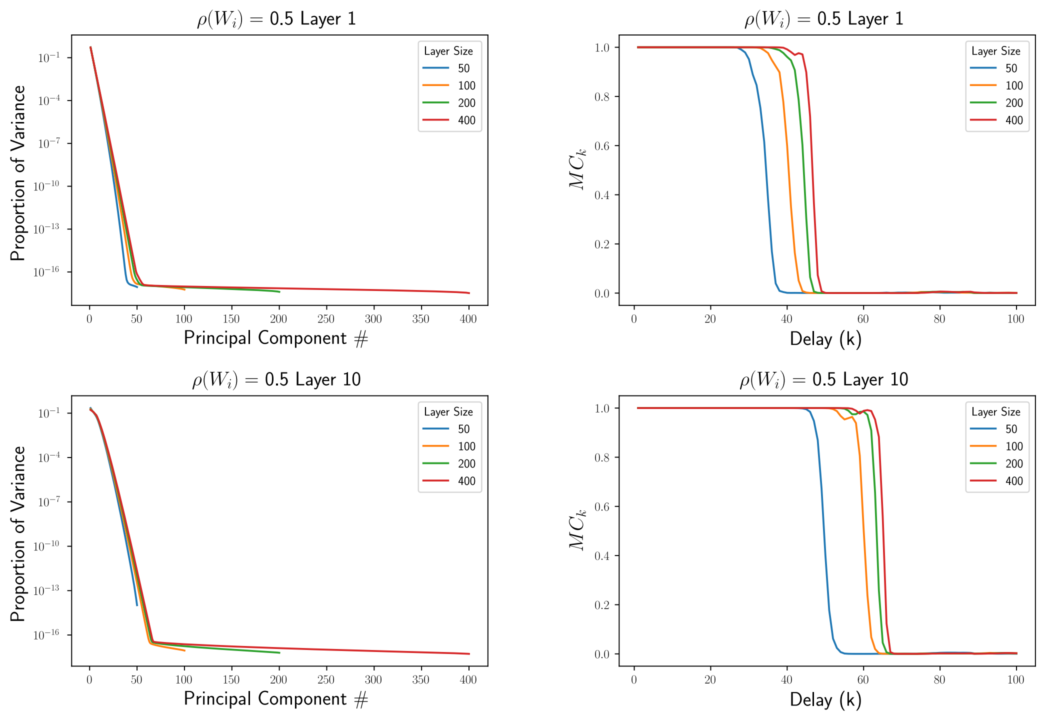
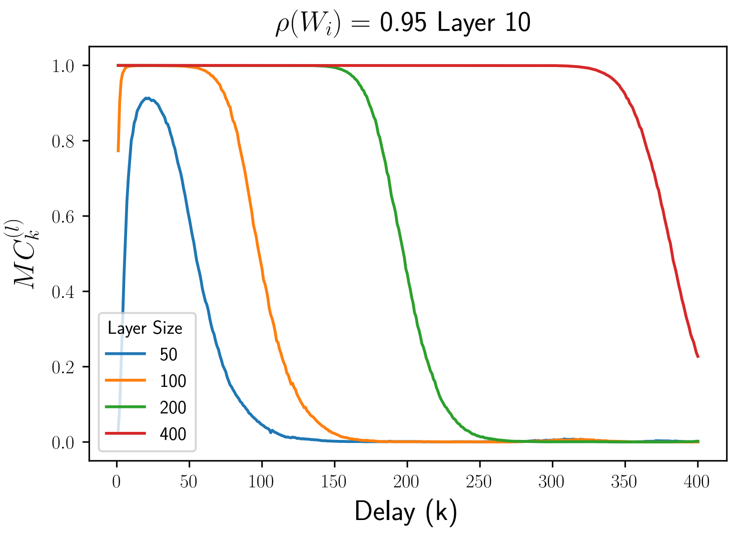


Figure 5.11: Left: Proportion of variance for which each principal component is responsible in the first and tenth layers of Lin-DeepESNs of varying sizes with spectral radius 0.5. Right: Memory capacity curves for those same network layers. Both figures show the average over 200 random network initialisations.



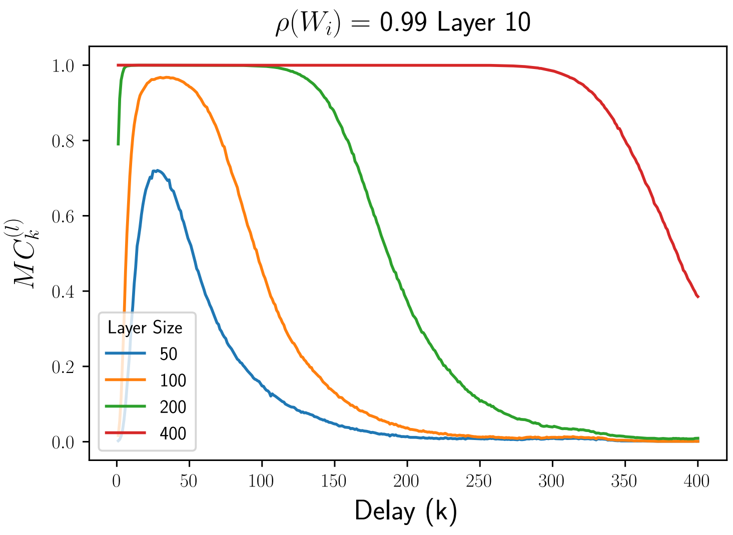


Figure 5.12: Memory capacity curves for the 10th layer of Lin-DeepESNs with spectral radii 0.95 and 0.99. For each layer size, we plot the average value of each MC_k over 200 random initialisations

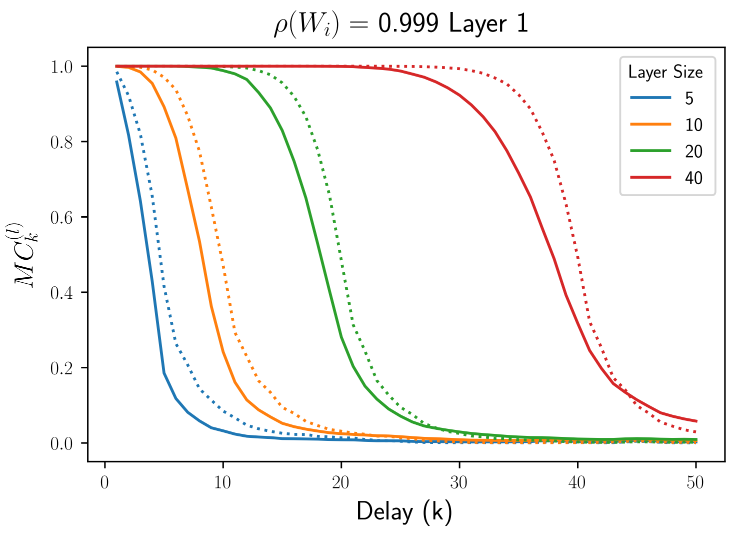


Figure 5.13: Memory capacity curve for the first layer of an Lin-DeepESN with spectral radius 0.999 averaged over 200 random initialisations (solid lines). The memory capacity curves for networks with spectral radius 0.9 are shown as dashed lines.

of how these networks are used in practice. In this section, we briefly examine how the effects we have observed in the linear case manifest differently when a saturating activation function is used.

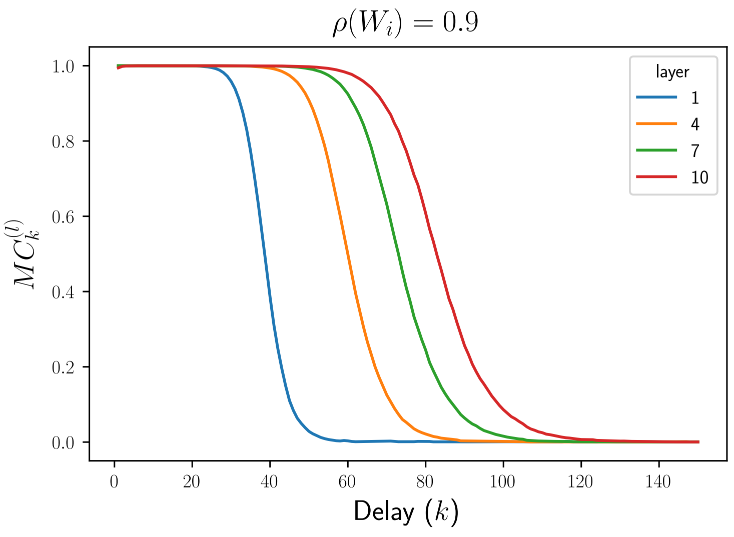
An important consideration when moving from the linear to the non-linear case is that we no longer have the scale-invariance which we benefited from in the linear case. In particular, as the norm of the components of the hidden state before the activation increases, so does the compressive effect of the non-linearity, and in this sense, the dynamics of the layer will be 'more non-linear'. In this section, we will show why it is an important to consider this behaviour when designing DeepESNs, and we will demonstrate the impact of both the recurrent spectral radius and the feedforward norm on this effect.

As a motivating example, we perform a similar experiment to that performed in [Gal18] measuring the memory capacity curves of various layers of a non-linear DeepESN. In particular, we examine the typical layer-wise memory capacity of a ten-layer network with 100 hidden units in each layer, using the tanh activation function. The weights in these networks are initialised using a uniform distribution over the interval $[-1, 1]$, then the weight matrices are rescaled to have the desired characteristics: each recurrent weight matrix is scaled so that its spectral radius is 0.9, feedforward weight matrices are scaled to have unit norm, and the input weight vector is scaled to have a norm of 0.1. We repeat the experiment 200 times and report the average of the results. For this experiment, and the rest of the experiments in this section, we use an input sequence of length 6000, with entries sampled independently from the interval $[-1, 1]$. We discard the first 100 network states, the next 4900 are used for training the regression weights and the last 1000 are used for calculating the memory capacity given those weights. The average of the memory capacity curves for this experiment is shown in Figure 5.14a.

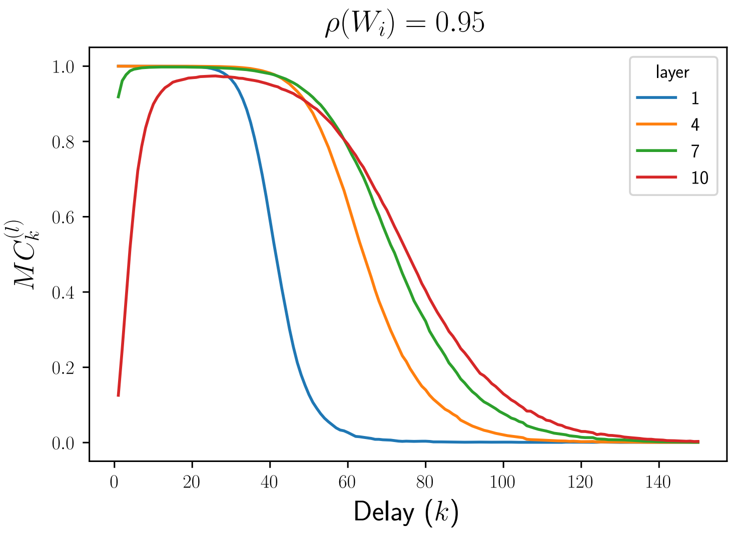
Like [Gal18], we observe that deeper network layers have higher memory capacity, and are able to retain information for longer. However, if we increase the spectral radius from 0.9 to 0.95, we get Figure 5.14b, where the last layer shows a substantial drop in memory capacity compared to the the case where the spectral radius is 0.9 (from a average total memory capacity of 83.03 to 71.15). While we also observe a drop in memory capacity in the 10th layer in the linear case, as shown in Table 5.3, the effect in the linear network is much smaller, both in terms of absolute value and as a proportion of the total memory capacity (dropping from 99.75 to 99.38).

If the feedforward weights are not sufficiently small as to compress the state of a layer before it is passed to the next layer, the average size of the hidden state grows at each layer. In the linear case, this would lead to exponential growth of the hidden state norm as we get deeper in the network; in the non-linear case, the non-linearity has an increasingly large effect at each layer. We hypothesise that this is the cause of the drop in performance on the memory capacity task.

In order to test this hypothesis, we first conduct experiments to examine the relationship between the average norm of the hidden state of each layer and the recurrent spectral radius and feedforward norm of the network. In particular, we vary the spectral radius of the recurrent weight matrices and the spectral norm of the feedforward matrices, and examine what values of these lead to contraction between layers versus expansion. We generate 200 random networks, scale the recurrent and feedforward weight matrices, then run each network on a sequence of 1000 scalar values sampled uniformly from the interval $[-1, 1]$. The states of each layer of the network are recorded at each time-step, and the average norm of the last 900 states of each layer are recorded. The results of these experiments are shown in Figure 5.15. The intersection of the dotted and solid lines in Figure 5.15 shows when the feedforward norm becomes sufficiently large that we start seeing expansion rather than contraction in the average state size between layers. It



(a)



(b)

Figure 5.14: Memory capacity curves of DeepESNs. (a) shows the memory capacity curves of layers when $\rho(W_i) = 0.9$ for each i. (b) shows the memory capacity curves of layers when $\rho(W_i) = 0.95$

can be observed that increasing the spectral radius decreases the minimum size at which the feedforward weights cause expansion in the size of the hidden state between layers as opposed to contraction.

We now proceed to examine the relationship between this phenomenon and the layer-wise memory capacity of the network. Again, we use 200 randomly generated networks, rescaling the spectral radius and feedforward norms to the desired values. As in the previous memory capacity experiments in this section, we run the network on a sequence of 6000 inputs uniformly sampled from $[-1, 1]$, with the same 100/4900/1000 split for transient/train/test. As a practical matter, in conducting these experiments we found it necessary to rescale the first entry of $\mathbf{z}_t$ (i.e., the entry equal to u_t) so that it had the same variance within the training data as a typical entry of $\mathbf{x}_t$. Our failure to do so in preliminary runs lead to numerical issues where we would observe a degradation of memory capacity not just for excessively large values of the feedforward norm, but also for excessively small ones.

The results from the experiments are shown in Figure 5.16. Additionally, we conduct the same experiment with input weight norms $10^{-6}, 10^{-4}$ and 10^{-2}. The results of these experiments are shown in Appendix C in Figure C.1, Figure C.2 and Figure C.3, respectively. Once again, there are several aspects to these Figures which are worthy of discussion:

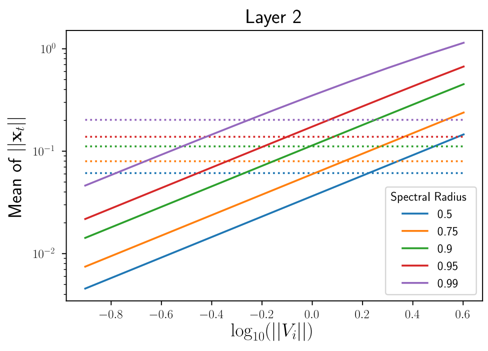
- **Behaviour for small values of $\|V_i\|$**: For small values of the feedforward norm, we see that the curves are relatively flat. As $\|V_i\| \to 0$, the inputs to each layer become closer to zero, and therefore the layer operates in the approximately linear region of the tanh function. This means that below a certain threshold, decreasing the feedforward weights further has little impact on the memory capacity. Note that the decision to scale $\|\mathbf{v}_i\|$ independently means that the maximum memory capacity reached by each layer is still significantly less than observed in the linear case in Table 5.3, likely because all signals to deeper layers must pass through the first layer, where the state is large enough for the non-linearity to have a detrimental effect. This can be verified by examining Figure C.1, where the input weights are scaled so that $\|\mathbf{v}_i\| = 10^{-6}$, and the memory capacities achieved are much closer to those observed in the linear case.

- **Effects of spectral radius**: Much like in the linear case, we see that in deeper layers of the network, it is not the case that larger spectral radius always implies a greater memory capacity. The reasons for this are likely the same reasons as discussed in the previous section. However, the non-linearity of the networks adds some complexity and we observe that the best spectral radius for a given layer size and depth can vary with the choice of feedforward norm.

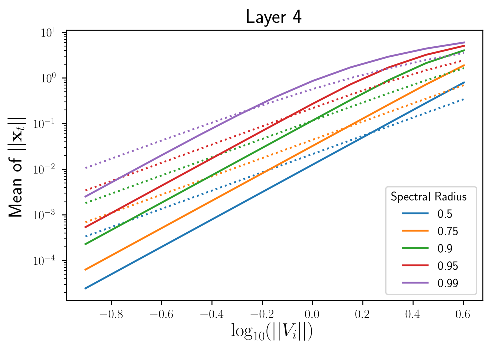
- **Contractive vs. expansive effects of $\|V_i\|$:** Noticeable memory degradation in all observed layers of our networks begins well before the point that the feedforward weight matrices go from being contractive to expansive. However, this is likely due in part to our relatively large input weights $\mathbf{v}_1$, since even with the input to layers getting smaller with depth, their states can still be large enough to be significantly affected by tanh's saturating effect. Indeed in Figures C.2 and C.1, we observe the opposite behaviour, especially in early layers of the network, i.e., even for values of $\|V_i\|$ which increase the size of states at between layers we do not observe a drop in memory capacity. Again, this makes intuitive sense: for extremely small $\|\mathbf{v}_1\|$, the signal has to be expanded over more applications of feedforward weight matrices until it encounters enough of an effect from the non-linearity in order for memory degradation to happen. While the contractive/expansive nature of the feedforward norms does appear to play an important part in determining the memory capacity of network layers, these results suggest our initial hypothesis was an over-simplification and fails to account for the full complexity of the system.

The effects that we observe here highlight an important shortfall in our current understanding of deep recurrent networks. In deep feedforward networks, principled methods for initialisation exist; these methods focus on initialising the weights so that the norms of the hidden states of each layer are the same size on average, as are the norms of the gradients under backpropagation [GB10; He+15][4]. These existing initialisation techniques do not naturally extend to deep recurrent networks, since the relationship between the variance of hidden states of different layers of a recurrent network and the network's weights is significantly more complex.

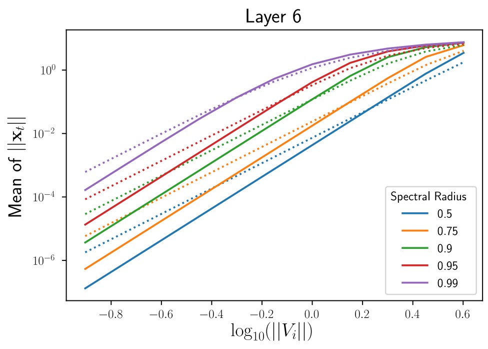
[4]It is interesting to note that for square matrices the scheme proposed in [GB10] is equivalent to setting the spectral radius to be close to 1. Indeed, we can think of initialising W to have spectral radius close (but strictly less than) to 1 in ESNs as having the same motivation as Glorot initialisation, i.e, preserving the magnitude of signal passing through W (even if in the latter case we do want eventual decay in order to preserve the ESP).



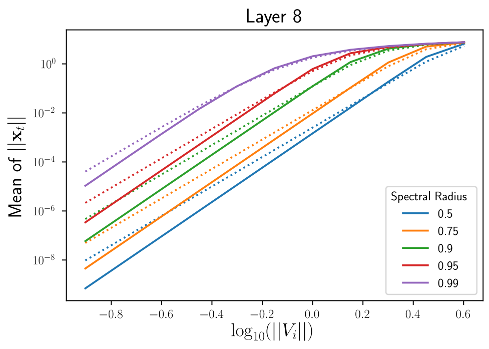
(a)



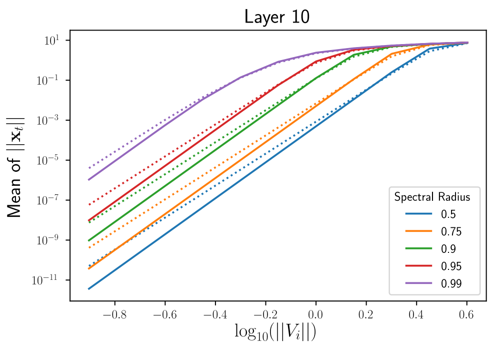
(b)



(c)

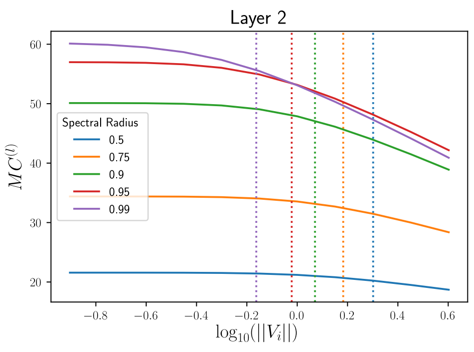


(d)

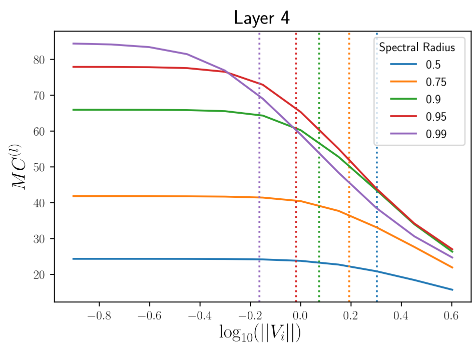


(e)

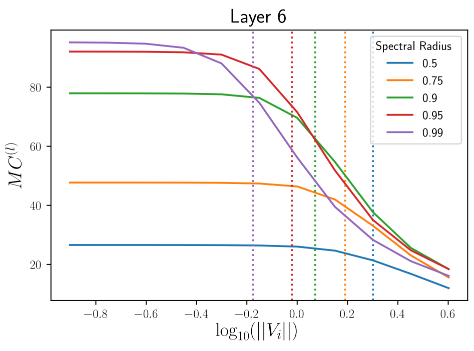
Figure 5.15: Mean size of state of hidden layer in non-linear network against the size of the norm of the feedforward weight matrices. Solid lines show the norm of the layer under consideration and dotted lines show the mean of the norm of the previous layer.



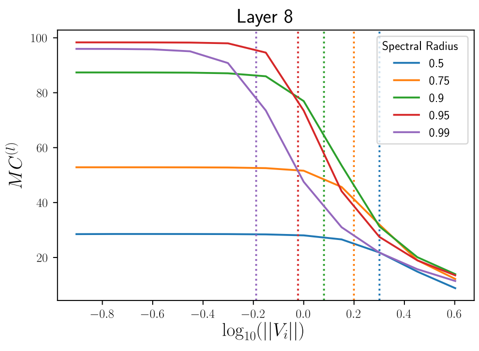
(a)



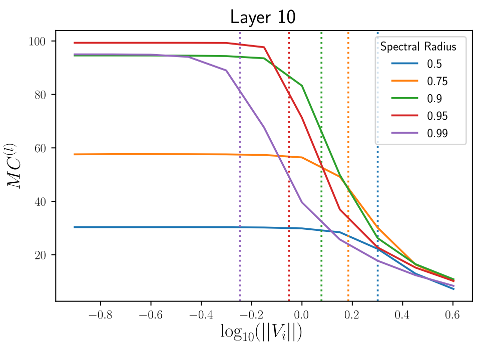
(b)



(c)



(d)



(e)

Figure 5.16: Total memory capacity for network layers, varying recurrent spectral radius and feedforward norm size. Input weight vector scaled to satisfy $\|\mathbf{v}_i\| = 0.1$ in all cases. Vertical lines show the feedforward norm at which the average norm of the current layer's state is larger than the average norm of the previous layer's state.

5.6 Conclusion

In this chapter, we have examined the different time-scales phenomenon in deep recurrent networks.

We have shown the phenomenon is exhibited in linear networks, despite the fact that perturbations in such networks necessarily exhibit asymptotically exponentially decreasing influence. We have further shown that the first layer of the such networks is qualitatively different in its behaviour compared to deeper layers in the network. By confirming the existence of the phenomenon in linear networks, we were able to study it in more detail than existing work (e.g., [GM16]), giving deeper insight into the nature of the phenomenon.

We provided a similar contribution for memory capacity, not only showing that the phenomenon previously observed for non-linear networks exists in the linear case, but also highlighting its complex relationship to the spectral radius of the recurrent weight matrix. Furthermore, we proved that the difference in memory capacity between layers in the linear case is due to the finite precision with which the computation was performed. Using this knowledge, we were able to more deeply investigate the phenomenon.

Finally, we demonstrated the importance of careful scaling of the feedforward matrix norms in the non-linear case, showing that improper scaling leads to a degradation in performance in the memory capacity task. We highlight the problem of managing these effects as a potential area of future research.

Most of the results in this chapter have been empirical in nature. While this research provides important insights into the causes of previously observed phenomena, it does not offer complete explanations for why this behaviour occurs. To the best of our knowledge, no work has been done on understanding these systems from a random matrix theory perspective, and we believe that this could be a productive area of future research.

By increasing the understanding of the nature of the phenomenon, we hope to pave the way for future research into how this phenomenon can be exploited to allow for improved performance of networks in tasks which require the network to approximate complex interactions between inputs in the recent and distant past.

Chapter 6

The Asymptotic Behaviour of Linear Input-Driven Systems

6.1 Introduction

When investigating the relationship between network weight structure and network behaviour, it is often useful to consider the case of linear networks (as, for example, we have done in previous chapters). However, removing the saturating non-linearity introduces new practical considerations in ESN construction. In particular, it is highly desirable that, given uniformly bounded inputs, the space of all reachable hidden state configurations is also bounded. This is trivial in the case where the state is subjected to a saturating activation function, but in the linear case, more care must be taken to ensure stability in this sense.

In the simplest terms, the relationship between the spectral radius of the recurrent matrix and the stability of the network is described by well-known results from linear systems theory (i.e., $\rho(W) \geq 1$ allows for the possibility that unstable trajectories through the state-space emerge from a network driven by bounded inputs, whereas $\rho(W) < 1$ does not). And indeed, most investigations into linear networks restrict their analyses to the case where the spectral radius of the recurrent weight matrix satisfies $\rho(W) < 1$.

While existing work will usually state that they consider only the case $\rho(W) < 1$, the connection to network stability is rarely made explicit. Instead, the restriction is sometimes framed in terms of ensuring the necessary condition for the Echo State Property (ESP) is met (e.g., [GGM18; GMP19]). It is worth remembering that in the non-linear case, $\rho(W) < 1$ is also necessary for the

ESP, but performance of the network can sometimes be improved by choosing $\rho(W) > 1$, hence, for the linear case it is more natural to think of $\rho(W) < 1$ implying stability as the reason for it being preferred[1]. However, it is rare to find work that explicitly discusses the conditions for stability of a linear ESNs—and even when the connection between the recurrent weight matrix and stability is made, it is often only as a passing comment, without referring the reader to a source where more detail can be found (e.g., [WLS04; OXP07]). Part of the motivation of this chapter is to bring knowledge relating the recurrent weight matrix's spectrum and stability inside of the ESN literature, rather than relying on the reader's existing knowledge of linear systems theory. In addition, we offer results regarding the consequences of network instability. We examine the effects an ESN's weights have on the asymptotic behaviour of the network, focusing on the relationship between a network's long-term behaviour and the spectrum of its recurrent weight matrix W (and also subject to some conditions on the input weight matrix V). In doing so, we provide a more complete understanding of how network structure influences the long-term behaviour of the network. We consider three distinct cases: the stable case ($\rho(W) < 1$), the unstable case ($\rho(W) = 1$), and the explosive case ($\rho(W) > 1$), allowing us to provide a more comprehensive understanding of the way that a network's weights affect the network's behaviour.

We note that though in the context of this thesis, it is most appropriate to frame the results in terms of linear ESNs, they are general enough to apply to a broader range of input-driven linear systems. Indeed, even though instability is typically an undesirable property in ESNs, unstable linear systems are of interest in other domains [KCP86; TT90; STM18]. [STM18] in particular, discusses several domains in which unstable linear systems arise and describes how auto-regressive models can be a useful tool in modelling those systems.

The contributions in this chapter are as follows:

- For linear ESNs driven by input from a compact set, we show that if $\rho(W) < 1$, then the network's hidden state is bounded and that the echo state property is satisfied. While the first of these is a standard result in linear systems theory, to the best of our knowledge it is yet to be explicitly included in the ESN literature. Furthermore, we show that light-tailed inputs

[1] Though $\rho(W) > 1$ means that the necessary condition for the ESP is not met, in the non-linear case large enough inputs can still result in the network becoming insensitive to initial conditions. While this adds nuance to discussion of non-linear case, it does not apply to linear networks.

guarantee that the distributions of the components of the network state are also light-tailed. In particular, for sub-Gaussian random variables (as defined later in this chapter in Section 6.2.1), we show that in the zero-mean case the distribution of the state vector is also sub-Gaussian, and that the distribution has sub-exponential tails in the case where the input has a deterministic component and additive sub-Gaussian noise.

- For the case $\rho(W) = 1$, we show that for independently distributed random inputs from zero-mean distributions, the expected rate of growth of the of the norm is polynomial in sequence length, with the degree of the polynomial determined by the structure of W and the directions of the columns of the input weight matrix V. We also extend this result to show the consequences for deterministic inputs perturbed by additive noise.

- For the case $\rho(W) > 1$, we show that the asymptotic rate of growth is exponential regardless of the structure of W, except for a simple condition on the relationship between W and V which ensures proper coupling of the input with the network state.

- The echo state property implies that the states of two identical networks with different initial conditions will converge asymptotically if the networks are driven by the same input. We show a complementary property: that in the case that identical networks are driven by two distinct sequences, the states of two networks will converge if and only if the two input sequences converge.

The rest of this chapter is organised as follows. In Section 6.2, we define the model and notation used throughout the rest of the chapter. In Sections 6.3, 6.4 and 6.5 we present our analysis of model behaviour in the three major cases: $\rho(W) < 1$, $\rho(W) = 1$ and $\rho(W) > 1$. In Section 6.6, we examine how the asymptotic behaviour of the network is determined by the asymptotic behaviour of the input. Finally, Section 6.7 contains some concluding remarks for the chapter.

6.2 Background

Throughout this chapter, we consider a linear ESN with M hidden units. In the case of single-dimensional input sequences, we denote the input sequence $\{u_t\}_{t=1}^{\infty}$

with $u_t \in U$, where U is a compact subset of the real line. In this case, we give the update rule of the network as

$$\mathbf{x}_t = W\mathbf{x}_{t-1} + \mathbf{v}u_t. \tag{6.1}$$

We refer to $W \in \mathbb{R}^{M\times M}$ as the recurrent weight matrix and $\mathbf{v} \in \mathbb{R}^M$ as the input weight vector, and we refer to the network itself as $(W, \mathbf{v})$.

In the multi-dimensional input case, we denote the input sequence $\{\mathbf{u}_t\}_{t=1}^{\infty}$, with $\mathbf{u}_t \in U$ where U is a compact subset of $\mathbb{R}^D$ for some $D \in \mathbb{N}$. In this case, the input weight vector is replaced with an input weight matrix $V \in \mathbb{R}^{M\times D}$ and the update rule becomes

$$\mathbf{x}_t = W\mathbf{x}_{t-1} + V\mathbf{u}_t. \tag{6.2}$$

Once again, we refer to the network using the notation (W, V). We define $\rho(W)$ to be the spectral radius of W, i.e., $\rho(W) \overset{\text{def}}{=} \max(\{|\lambda| : \lambda \in S(W)\})$, where $S(W)$ is the set of eigenvalues of W, and $\|W\|$ to be the spectral norm of W, i.e., $\|W\| \overset{\text{def}}{=} \max_{\mathbf{x}\in\mathbb{R}^M} \frac{\|W\mathbf{x}\|}{\|\mathbf{x}\|}$. Unless stated otherwise, we set the initial state of the network $\mathbf{x}_0 = \mathbf{0}$, though all the results stated are easily modifiable to hold for non-zero initial state.

In this chapter we are interested in the asymptotic behaviour of these systems, and in particular whether or not the systems are stable. For this reason, it is useful to formalise the notion of stability as follows:

Definition 6.2.1. *Consider a dynamical system with some update function $F(\cdot,\cdot)$, such that $\mathbf{x}_t = F(\mathbf{x}_{t-1}, \mathbf{u}_t)$, where $\{\mathbf{u}_t\}_{t=1}^{\infty}$ is a sequence of vectors in some compact subset of $\mathbb{R}^D$, U. We call the system* stable*, if there exists a constant C, such that for all permitted input sequences, $\|\mathbf{x}_t\| < C$ for all $t \in \mathbb{N}$. Conversely, we call the system unstable if there exists an input sequence $\{\mathbf{u}_t\}_{t=1}^{\infty}$ such that for all C, there exists t such that $\|\mathbf{x}_t\| > C$.*

It is worth noting that in general neither the echo state property nor stability imply the other. In one direction, we just need to note the existence of networks with tanh as their activation function where the echo state property doesn't hold (see, for instance, [YJK12]). For the other direction, we can imagine constructing a system where all trajectories converge to be arbitrarily close to a single trajectory whose norm grows without limit, satisfying the ESP without being stable (as a trivial example, we could consider the 1-dimensional system $f(x_t, u_t) = t$).

In what follows, it will frequently be convenient to consider vectors in the basis of generalized eigenvectors of W. In order to do so, we fix some notation here. Firstly, we define the Jordan decomposition of W as $W = PJP^{-1}$, where J is the Jordan normal form of W and P is the matrix whose columns are the generalized eigenvectors of W, ordered to correspond with the entries of J in the usual manner. We define the vector norm $\|\cdot\|_P$ as the norm satisfying $\|\mathbf{x}\|_P \overset{\text{def}}{=} \|P^{-1}\mathbf{x}\|_\infty$ for all $\mathbf{x} \in \mathbb{R}^M$, i.e., the infinity norm[2] of the vector in the basis of generalized eigenvectors of W.

If we have a vector of the form $W^k\mathbf{v}u$, the component in the direction of the ith generalized eigenvector can be written as

$$\left(P^{-1}W^k\mathbf{v}u\right)_i = \left(J^kP^{-1}\mathbf{v}u\right)_i.$$

Using the expression for powers of Jordan blocks in Appendix A, if the ith component is in the Jordan block of size d' running from j to $j + d' - 1$, and we write $\mathbf{v}' \overset{\text{def}}{=} P^{-1}\mathbf{v}$, then the component is given by

$$\left(P^{-1}W^k\mathbf{v}u\right)_i = \sum_{l=i}^{j+d'-1} \binom{k}{l-i} \cdot \lambda_n^{k-(l-i)} \cdot \mathbf{v}'_l \cdot u. \tag{6.3}$$

Note that we can write the magnitude as

$$\left|\left(P^{-1}W^k\mathbf{v}u\right)_i\right| = \left|\lambda_n^k\right| \cdot \left|\sum_{l=i}^{j+d'-1} \binom{k}{l-i} \cdot \lambda_n^{i-l} \cdot \mathbf{v}'_l \cdot u\right|, \tag{6.4}$$

and in the case where $|\lambda_n| = 1$, we can eliminate the first term on the right hand side. We define the function

$$p_i(k) \overset{\text{def}}{=} \sum_{l=i}^{j+d'-1} \binom{k}{l-i} \cdot \lambda_n^{(i-l)} \cdot \mathbf{v}'_l, \tag{6.5}$$

and note that by expansion of the combination function, we can show that each $p_i(k)$ is a polynomial of degree at most $d' - 1$ [Lea94], and we can write

$$\left(P^{-1}W^k\mathbf{v}'u\right)_i = \lambda_i^k \cdot p_i(k) \cdot u. \tag{6.6}$$

[2]The infinity norm on a vector $\mathbf{x} \in \mathbb{R}$ is defined as $\|\mathbf{x}\|_\infty = \max(\{|(\mathbf{x})_i| : 1 \leq i \leq M\}$

Of particular use is the fact that when $|\lambda_n| = 1$,

$$\left|\left(P^{-1}W^k\mathbf{v}'u\right)_i\right| = |p_i(k) \cdot u| \,. \tag{6.7}$$

Throughout this chapter, we will will consider linear networks driven by both deterministic input sequences and sequences of random variables. In the latter case, the network state is a random vector. In order to distinguish between the two cases, we denote the hidden state of a network driven by random inputs as $\boldsymbol{x}_t$, in contrast with the deterministic state $\mathbf{x}_t$.

To aid our analysis, we will make frequent use of the following result from linear algebra. We provide proof of the result in Appendix A, but also state the result here for clarity.

Lemma A.3.1. *Let A be a matrix such that $\rho(A) < 1$. For all γ satisfying $\rho(A) < \gamma$, there exists $k_0 \in \mathbb{N}$ such that, for all $k > k_0$, $\left\|A^k\right\| < \gamma^k$.*

This result is useful in considering the asymptotic behaviour of a network, as combining it with the sub-multiplicity of the norm not only gives that the contribution of an input k time-steps in the past eventually converges to zero[3], but that it eventually converges at an exponential rate. Unfortunately the result is not strong enough to tell us anything about the short term behaviour of the network, since for small k, not only can $\left\|W^k\right\|$ be significantly greater than the spectral radius, we also don't have guarantees that it is monotonically decreasing.

In addition to the material presented here, this chapter relies on several well-known inequalities from linear algebra and probability theory. These are referred by name when on first use, and are listed, along with references, in Appendix A.2.

6.2.1 Sub-Gaussian Random Variables

In the case where $\rho(W) < 1$, we explore what happens when we remove the usual restriction of taking inputs from a compact subset of a real coordinate space and instead impose the restriction that the inputs are taken from a distribution with sufficiently light tails. In particular, we examine the case where the input distributions are *sub-Gaussian*. In this sub-section, we define what it means for a variable to be sub-Gaussian, and state some useful results relating to sub-Gaussian variables.

[3]Since we can write $\left\|W^k\mathbf{v}u_{t-k}\right\| \leq \left\|W^k\right\| \cdot \left\|\mathbf{v}u_{t-k}\right\|$

Definition 6.2.2 (Sub-Gaussian Random Variable)**.** *Let $\boldsymbol{x}$ be a random vector with support in $\mathbb{R}^M$. We say that $\boldsymbol{x}$ is sub-Gaussian with proxy variance σ^2 if $\mathbb{E}[\boldsymbol{x}] = 0$ and for all unit vectors $\mathbf{u} \in S^{M-1}$, and all $s \in \mathbb{R}$*

$$\mathbb{P}(|\mathbf{u}^\intercal \boldsymbol{x}| > s) \leq 2 \cdot \exp\left(\frac{-s^2}{2 \cdot \sigma^2}\right)$$

Note that in the case where $\boldsymbol{x}$ is a random variable taking values in $\mathbb{R}$, the condition simplifies to $\mathbb{P}(|\boldsymbol{x}| > s) \leq 2 \cdot \exp\left(\frac{-s^2}{2 \cdot \sigma^2}\right)$.

While a sub-Gaussian random variable may have non-zero density on the entire space, the tails of the distribution drop off at least as fast as the tails of a Gaussian distribution. This means that while it is possible for arbitrarily large values to be observed, they are extremely improbable. The Gaussian, in both its univariate and multivariate forms, is a notable example of a sub-Gaussian distribution, so the results for sub-Gaussian inputs are directly applicable to networks where the input is subjected to additive Gaussian noise.

An equivalent definition of a sub-Gaussian random variable is given by the following lemma.

Lemma 6.2.1. *A random variable $\boldsymbol{x}$ with support in $\mathbb{R}^M$ is sub-Gaussian with proxy variance σ^2 if and only if $\mathbb{E}[\boldsymbol{x}] = 0$ and for all $t \in \mathbb{R}$, and all $\mathbf{u} \in S^{M-1}$*

$$\mathbb{E}\left[\exp\left(t \cdot \mathbf{u}^\intercal \boldsymbol{x}\right)\right] \leq \exp\left(\frac{\sigma^2 \cdot t^2}{2}\right).$$

Proof. See, for instance [Riv12]. □

We state this lemma because it provides an alternative definition for sub-Gaussian random variables and vectors which can be used to trivially prove the following two properties:

1. If $\boldsymbol{x}$ is a sub-Gaussian with proxy variance σ^2, for any $\alpha \in \mathbb{R}$, $\alpha\boldsymbol{x}$ is sub-Gaussian with proxy variance $\alpha^2 \cdot \sigma^2$

2. If $\boldsymbol{x}$ is sub-Gaussian with proxy variance σ_1^2, for any $\sigma_2^2 > \sigma_1^2$, $\boldsymbol{x}$ is sub-Gaussian with proxy variance σ_2^2.

We will also use the following lemma concerning the sum of sub-Gaussian random variables.

Lemma 6.2.2. *Let $X_1, \dots X_n$ be n independent scalar (i.e, 1-dimensional) sub-Gaussian random variables, where X_i has proxy variance σ_i^2. For all $s > 0$*

$$\mathbb{P}\left[\left|\sum_{i=1}^{n} X_i\right| > s\right] \leq 2 \cdot \exp\left(\frac{-s^2}{2 \cdot \sum_{i=1}^{n} \sigma_i^2}\right)$$

Proof. See, for instance [Wai15]. □

6.3 The Stable Case: $\rho(W) < 1$

In this section, we examine the behaviour of linear echo state networks when $\rho(W) < 1$. In the literature, analysis of ESNs is usually restricted to this case, with the implicit assumption that this ensures network stability; in this section, we show explicitly why this is the case, and that in linear networks $\rho(W) < 1$ serves as a necessary and sufficient condition for the echo state property. Furthermore, we show that even when the input is perturbed by additive noise, we retain stability in the sense that sub-Gaussian noise leads to the distribution of the norm of the hidden state having sub-exponential tails (and sub-Gaussian tails in the case where the input has no deterministic component).

In linear systems theory, the relationship between the norm of the hidden state and the spectral radius of the recurrent matrix is a well understood subject. Indeed, the stability of linear echo state networks under bounded inputs is just a special case of external stability (or bounded-input bounded-output stability) in linear systems theory. Proof of the more general case can be found in linear systems textbooks (e.g. [Rug96]). However, for the sake of clarity and completeness, we provide a direct proof for the specific network structure in Equation 6.2.

Theorem 6.3.1. *Given an echo state network with recurrent weight matrix W such that $\rho(W) < 1$ and inputs from a compact domain U, the hidden state of the network is bounded, i.e., there exists a constant C_{state} such that for all right-infinite input sequences $\{\mathbf{u}_t\}_{t=1}^{\infty}$ with each $\mathbf{u}_t \in U$ and for all t, $\|\mathbf{x}_t\| < C_{state}$.*

Proof. We define $C_{in} \overset{\text{def}}{=} \max_{\mathbf{u} \in U} \|V\mathbf{u}\|$ and examine the norm of the hidden state using the triangle inequality and sub-multiplicativity of the norm,

$$\begin{aligned}
\|\mathbf{x}_t\| &= \left\| \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} \right\| \\
&\leq \sum_{k=0}^{t-1} \left\| W^k V \mathbf{u}_{t-k} \right\| \\
&\leq \sum_{k=0}^{t-1} \left\| W^k \right\| \cdot \| V \mathbf{u}_{t-k} \| \\
&\leq C_{in} \cdot \sum_{k=0}^{t-1} \left\| W^k \right\| .
\end{aligned}$$

We now use Lemma A.3.1, choosing γ satisfying $\rho(W) < \gamma < 1$ and define k_0 as the smallest natural number such that for all $k \geq k_0$, $\left\| W^k \right\| < \gamma^k$. Next, we define C_0 as an upper bound on the value the norm of the contributions from inputs in the k_0 most recent time-steps:

$$C_0 = C_{in} \cdot \sum_{k=0}^{k_0 - 1} \left\| W^k \right\| .$$

We now consider the maximum size the norm of the hidden state. For any $t \geq k_0$, we have

$$\begin{aligned}
\|\mathbf{x}_t\| &\leq C_{in} \cdot \sum_{k=0}^{t-1} \left\| W^k \right\| \\
&\leq C_{in} \left(\sum_{k=0}^{k_0-1} \left\| W^k \right\| + \sum_{k=k_0}^{t-1} \left\| W^k \right\| \right) \\
&= C_0 + C_{in} \cdot \sum_{k=k_0}^{t-1} \left\| W^k \right\| \\
&= C_0 + C_{in} \cdot \sum_{k=k_0}^{t-1} \gamma^k \\
&< C_0 + \frac{C_{in} \cdot \gamma^{k_0}}{1-\gamma},
\end{aligned}$$

therefore the norm of the hidden state is bounded by $C_{state} \overset{\text{def}}{=} C_0 + \frac{C_{in} \cdot \gamma^{k_0}}{1-\gamma}$. $\square$

An oft cited misconception about ESNs is that spectral radius less than one is sufficient condition to ensure the echo state property [LJ09; YJK12; Cal+13; MJ13]. This misconception probably arises due to its apparent sufficiency in practice [Luk12; Cal+13]. Adding further nuance to this story, we show that $\rho(W) < 1$ is sufficient condition for the echo state property when the network is linear.

Theorem 6.3.2. *For linear networks with admissible initial states in the set $A = \{\mathbf{x} \in \mathbb{R}^M : \|\mathbf{x}\| < C\}$ for some positive constant C, $\rho(W) < 1$ is both sufficient and necessary condition for the echo state property.*

Proof. As mentioned in the previous section, $\rho(W) < 1$ is a well known necessary condition for the echo state property [Jae01], so we consider only whether it is also a sufficient condition. To show this, let $\mathbf{x}, \mathbf{x}' \in A$ be two network states. The distance at time t between two networks initialized to $\mathbf{x}$ and $\mathbf{x}'$ then driven by the same input sequence is:

$$\begin{aligned}
\|\mathbf{x}_t - \mathbf{x}'_t\| &= \left\| W^t\mathbf{x} + \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} - W^t\mathbf{x}' - \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} \right\| \\
&= \left\| W^t\mathbf{x} - W^t\mathbf{x}' \right\| \\
&\le \left\| W^t\mathbf{x} \right\| + \left\| W^t\mathbf{x}' \right\| \\
&\le \left\| W^t \right\| \cdot (\|\mathbf{x}\| + \|\mathbf{x}'\|) \\
&< 2 \cdot \left\| W^t \right\| \cdot C
\end{aligned}$$

Using Lemma A.3.1 we get that for any γ satisfying $\rho(W) < \gamma < 1$, and for some t_0, for all $t > t_0$, $\|W^t\| < \gamma^t$. Since $\lim_{t\to\infty} \gamma^t = 0$, we have $\lim_{t\to\infty} \|W^t\| = 0$. Consequently, the distance between the states of the two networks tends to zero, and since $\max_{\mathbf{x}\in A} \|\mathbf{x}\| = C < \infty$ the echo state property is satisfied. $\square$

This result means that in the linear case, we need not be concerned with the spectral norm—all the important information about the network's stability is contained within the spectral radius.

The echo state property describes only the asymptotic behaviour of the network; we suggest that it is likely that the above result can be extended to give non-asymptotic bounds on the difference between the network states—however, we leave this to future work.

6.3.1 Stable Networks with Sub-Gaussian Inputs

Thus far, we have restricted our analysis to the case where all inputs are drawn from a compact subset of a real space. However, this denies us the chance to consider sequences which are subjected to white noise. In order to expand our analysis to such inputs, we consider the situation where the input sequence is of the form $\{\mathbf{u}_i\}_{i=1}^{\infty}$, with each $\mathbf{u}_i$ decomposed into $\mathbf{u}_i = \hat{\mathbf{u}}_i + \boldsymbol{\epsilon}_i$ where each $\boldsymbol{\epsilon}_i$ is i.i.d. noise and each $\hat{\mathbf{u}}_i$ is drawn from the same compact subset of $\mathbb{R}^D$. Rather than specifying the distribution of the noise, the tails of the distribution are constrained to ensure that they are sufficiently light. In particular, we consider the case where the where the distribution is *sub-Gaussian*, as defined in Section 6.2.1.

Using the results from Section 6.2.1, we are able to show that if a linear echo state network is driven by a sequence of sub-Gaussian random variables, there is a proxy-variance σ^2 such that for all t, $\boldsymbol{x}_t$ is sub-Gaussian with proxy variance σ^2.

Theorem 6.3.3. *Let $\{\boldsymbol{\epsilon}_i\}_{i=1}^{\infty}$ be a sequence of D-dimensional sub-Gaussian random vectors each with proxy variance σ_0^2. Let $W \in \mathbb{R}^{M\times M}$ be a matrix satisfying $\rho(W) < 1$ and let $V \in \mathbb{R}^{M\times D}$. If the ESN (W, V) is driven by the input sequence $\{\boldsymbol{\epsilon}_i\}_{i=1}^{\infty}$, there exists proxy variance σ^2 such that for all t, $\boldsymbol{x}_t$ is sub-Gaussian with proxy variance σ^2*

Proof. Note that for any $\mathbf{u} \in S^{M-1}$, either $\left(\mathbf{u}^{\intercal}W^kV/\left\|\mathbf{u}^{\intercal}W^kV\right\|\right)^{\intercal} \in S^{d-1}$ or $\mathbf{u}^{\intercal}W^kV$ is the zero vector. If it's zero, trivially for all $s > 0$ and any $\sigma^2 > 0$,

$$\begin{aligned}\mathbb{P}\left(\left|\mathbf{u}^{\intercal}W^kV\boldsymbol{\epsilon}_{t-k}\right| > s\right) &= 0\\ &< 2\cdot\exp\left(\frac{-s^2}{2\cdot\sigma^2}\right),\end{aligned}$$

otherwise the fact that $\boldsymbol{\epsilon}_{t-k}$ is sub-Gaussian gives us,

$$\begin{aligned}\mathbb{P}\left(\left|\mathbf{u}^{\intercal}W^kV\boldsymbol{\epsilon}_{t-k}\right| > s\right) &= \mathbb{P}\left(\frac{\left|\mathbf{u}^{\intercal}W^kV\boldsymbol{\epsilon}_{t-k}\right|}{\left\|\mathbf{u}^{\intercal}W^kV\right\|} > \frac{s}{\left\|\mathbf{u}^{\intercal}W^kV\right\|}\right)\\ &= \mathbb{P}\left(\left|\frac{\mathbf{u}^{\intercal}W^kV}{\left\|\mathbf{u}^{\intercal}W^kV\right\|}\boldsymbol{\epsilon}_{t-k}\right| > \frac{s}{\left\|\mathbf{u}^{\intercal}W^kV\right\|}\right)\\ &\le 2\cdot\exp\left(\frac{\left(\frac{-s^2}{\left\|\mathbf{u}^{\intercal}W^kV\right\|^2}\right)}{2\cdot\sigma_0^2}\right)\end{aligned}$$

$$= 2 \cdot \exp\left(\frac{-s^2}{2 \cdot \|\mathbf{u}^\intercal W^k V\|^2 \cdot \sigma_0^2}\right)$$
$$\leq 2 \cdot \exp\left(\frac{-s^2}{2 \cdot \|W^k V\|^2 \cdot \sigma_0^2}\right),$$

so for any k, $W^k V \boldsymbol{\epsilon_{t-k}}$ is sub-Gaussian with proxy variance $\left\|W^k V\right\|^2 \cdot \sigma_0^2$ By Lemma A.3.1, we have that there exist constants $\rho(W) < \gamma < 1$ and $C' > 0$ such that for all non-negative integers k, $\left\|W^k\right\| < C' \cdot \gamma^k$, and so $\left\|W^k V\right\| \leq \left\|W^k\right\| \cdot \|V\| < C \cdot \gamma^k$, where $C \overset{\text{def}}{=} C' \cdot \|V\|$. This gives that $W^k V \boldsymbol{\epsilon_{t-k}}$ is sub-Gaussian with proxy variance $C^2 \cdot \gamma^{2 \cdot k} \cdot \sigma_0^2$ for all $k \in \mathbb{N}$.

Now we can use Lemma 6.2.2 and the fact $\sum_{k=0}^{t-1} \gamma^{2 \cdot k} < \frac{1}{1-\gamma^{2 \cdot k}}$ to get that for all $\mathbf{u} \in \mathbb{S}^{M-1}$ and all t,

$$\mathbb{P}(|\mathbf{u}^\intercal \boldsymbol{x}_t| > s) = \mathbb{P}\left(\left|\mathbf{u}^\intercal \left(\sum_{k=0}^{t-1} W^k V(\boldsymbol{\epsilon}_{t-k})\right)\right| > s\right)$$
$$= \mathbb{P}\left(\left|\left(\sum_{k=0}^{t-1} \mathbf{u}^\intercal W^k V \boldsymbol{\epsilon}_{t-k}\right)\right| > s\right)$$
$$\leq 2 \cdot \exp\left(\frac{-s^2}{\sum_{k=0}^{t-1} C^2 \cdot \gamma^{2 \cdot k} \cdot \sigma_0^2}\right)$$
$$< 2 \cdot \exp\left(\frac{-s^2}{\frac{C^2 \cdot \sigma_0^2}{1-\gamma^2}}\right).$$

Therefore, for all t, $\boldsymbol{x}_t$ is sub-Gaussian with proxy variance $\frac{C^2 \cdot \sigma_0^2}{1-\gamma^2}$. □

In particular, this gives us strong concentration bounds on the components of the basis vectors of $\mathbb{R}^M$, so components in these directions are unlikely to be extremely large (we could trivially use the union bound to bound the probability that *any* component of $\boldsymbol{x}_t$ in the direction of a basis vector is above a given threshold at the expense of a factor of M).

Thus far, we have considered only the case where the network is driven by zero mean sub-Gaussian noise. However, this is not a situation which is likely to occur in practice. A more common scenario is that our network is driven by an input signal which is deterministic, but perturbed by additive noise. In this case, the state of the network is not necessarily sub-Gaussian, even if the noise is. Though we could re-centre around the mean state at time t and show that this re-centred state is sub-Gaussian, this is not particularly useful, as we are interested in the asymptotic behaviour of the distribution of the size of entries of $\boldsymbol{x}_t$, not the asymptotic distance from the mean. In order to better understand the behaviour in this case, we present the following theorem.

Theorem 6.3.4. *Let (W, V) be an ESN with $\rho(W) < 1$. Let $\{\boldsymbol{\epsilon_t}\}_{t=1}^{\infty}$ be a sequence of sub-Gaussian random vectors, each with proxy variance σ_0^2, and U be a compact subset of $\mathbb{R}^D$. For all $1 \le \beta < 2$, there exist constants $s_0 \ge 0$ and $\sigma > 0$, such that for all sequences $\{\boldsymbol{u}_t\}_{t=1}^{\infty}$ where for each t, $\boldsymbol{u}_t = \mathbf{u}_t + \boldsymbol{\epsilon_t}$ with $\mathbf{u}_t \in U$, and for all t and $s > s_0$, when (W, V) is driven by $\{\mathbf{u}_t\}_{i=1}^{\infty}$, for all $\mathbf{u} \in S^{M-1}$*

$$\mathbb{P}(\mathbf{u}^\intercal \boldsymbol{x}_t > s) < 2 \cdot \exp\left(\frac{-s^\beta}{2 \cdot \sigma^2}\right).$$

Proof. We can define the state at time t, $\boldsymbol{x}_t$, as the sum of $\mathbf{x}'_t$ and $\boldsymbol{y}_t$, where $\mathbf{x}'_t = \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k}$ and $\boldsymbol{y}_t = \sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}$. By Theorem 6.3.3 for all t, $\boldsymbol{y}_t$ is sub-Gaussian with some proxy variance σ^2. This gives that for all $\mathbf{u} \in S^{M-1}$ and all $s > |\mathbf{u}^\intercal \mathbf{x}'_t|$

$$\begin{aligned}
\mathbb{P}\left(|\mathbf{u}^\intercal \boldsymbol{x}_t| > s\right) &= \mathbb{P}\left(|\mathbf{u}^\intercal \mathbf{x}'_t + \mathbf{u}^\intercal \boldsymbol{y}_t| > s\right) \\
&\le \mathbb{P}\left(|\mathbf{u}^\intercal \mathbf{x}'_t| + |\mathbf{u}^\intercal \boldsymbol{y}_t| > s\right) \\
&= \mathbb{P}\left(|\mathbf{u}^\intercal \boldsymbol{y}_t| > s - |\mathbf{u}^\intercal \mathbf{x}'_t|\right) \\
&\le 2 \cdot \exp\left(\frac{-(s - |\mathbf{u}^\intercal \mathbf{x}'_t|)^2}{2 \cdot \sigma^2}\right). \qquad (6.8)
\end{aligned}$$

By Theorem 6.3.1, we know that there exists $C_{state} > 0$ such that $\|\mathbf{x}'_t\| < C_{state}$ for all t and therefore $|\mathbf{u}^\intercal \mathbf{x}'_t| < C_{state}$. Define $\mu_t \overset{\text{def}}{=} \mathbf{u}^\intercal \mathbf{x}'_t$. We have

$$\begin{aligned}
(s - |\mu_t|)^2 &= s^2 - 2 \cdot |\mu_t| \cdot s + |\mu_t|^2 \\
&\ge s^2 - 2 \cdot |\mu_t| \cdot s \\
&\ge s^2 - 2 \cdot C_{state} \cdot s.
\end{aligned}$$

Eventually, $s^2 - 2 \cdot C_{state} \cdot s$ dominates s^β for any $1 \le \beta < 2$. Therefore, there exists s_0 such that for all possible non-negative $|\mu_t|$ and $s > s_0$,

$$(s - |\mu_t|)^2 > s^\beta,$$

for all such s, Equation 6.8 gives

$$\mathbb{P}(|\mathbf{u}^{\intercal}\mathbf{x}_t| > s) \leq \exp\left(\frac{-(s - |\mu_t|)^2}{2 \cdot \sigma^2}\right) < \exp\left(\frac{-s^{\beta}}{2 \cdot \sigma^2}\right),$$

giving the result. □

Though in the case of zero mean inputs (or a deterministic input sequence perturbed by Gaussian noise), the tails are not necessarily sub-Gaussian, they are sub-exponential, and therefore the probability of extreme values in the entries of the hidden state decays faster than exponentially in the tails.

6.4 The Unstable Case: $\rho(W) = 1$

We now consider the case $\rho(W) = 1$. In this case, we know that we cannot achieve the echo state property [Jae01], and from linear systems theory we know that unstable trajectories exist [Rug96] (i.e., there are input sequences for which the norm of the hidden state has no finite bound). In this section, we show that if the network is driven by a sequence of independent random vectors with zero mean and a common non-zero variance, then the expectation of the norm of the hidden state grows polynomially with time, and and we show how the degree of this polynomial can be determined by the structure of the network's weights.

Initially, we restrict ourselves to the case of 1-dimensional input, where the input sequence consists of 1-dimensional random variables ϵ_t, where for each t, ϵ_t is drawn independently from the same zero-mean distribution (or at a minimum—independent distributions with the same finite variance), though we will show how the result generalised to higher dimensional inputs later in the section.

We will soon find that the rate of growth of $\mathbb{E}\left[\|\boldsymbol{x}_t\|\right]$ is intimately linked to the Jordan canonical form of W and the direction of $\mathbf{v}$. In particular, we will find that it can be diminished if there are generalised eigenvectors of W which are orthogonal to $\mathbf{v}$. In order for us to be able to discuss this more precisely, we give the following definition.

Definition 6.4.1 (Degree of Alignment). *Let $\mathbf{v}$ be a vector in $\mathbb{R}^M$ and W a matrix in $\mathbb{R}^{M \times M}$. We say that $\mathbf{v}$ is aligned to W with degree d if d is the largest natural number such that there exists at least one generalized eigenvector $\mathbf{p}$ of W which: is of rank d, is not orthogonal to $\mathbf{v}$, and has corresponding eigenvalue λ such that $|\lambda| = \rho(W)$.*

We can see that if $\mathbf{v}$ is aligned to W with degree d and $\rho(W) = 1$, then by Equation 6.7 there exists a component of $P^{-1}W^k\mathbf{v}'$ whose magnitude grows with k at a rate polynomial in k with degree $d-1$. Note that there is a relationship between the notion of degree of alignment and effective rank from Chapter 4, in that the degree of alignment is the largest effective rank amongst Jordan blocks with eigenvalues equal in magnitude to the spectral radius of W. In what follows, we will assume that $\mathbf{v}$ is aligned to W with degree at least one. If we were to assume otherwise, then we could find a lower-dimensional system with a smaller spectral radius but the same representational power, which would mean that the system is in fact stable and we could instead apply the results from the previous section.

With the above definition, we are able to use the polynomial rate of growth of the components of the hidden state to get the following lemma.

Lemma 6.4.1. *Let W be a matrix such that $\rho(W) = 1$ and $\mathbf{v}$ a vector which is aligned to W with degree $d \geq 1$. Define $\boldsymbol{v}_k = \mathbf{v}\epsilon_k$, where each ϵ_k is a random variable with zero-mean and the same non-zero finite variance. There exist constants $C_1, C_2, t_0 > 0$ such that for all $t > t_0$,*

$$C_1 \cdot t^{2\cdot d-1} < \mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] < C_2 \cdot t^{2\cdot d-1}.$$

The proof of this lemma is provided in Appendix D.1.

If a function $f(t)$ is eventually bounded above and below by some constant multiples of another function $g(t)$, we will write $f(t) = \Theta(g(t))$. Using this notation, we can express the above lemma as $\mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}\right\|^2\right] = \Theta(t^{2\cdot d-1)})$.

Theorem 6.4.1. *Let $\boldsymbol{x}_t$ be the state of an ESN with update rule as in Equation 6.1 driven by the random input sequence $\{\epsilon_t\}_{t=1}^{\infty}$, with each ϵ_t having the same non-zero finite variance. Let $\rho(W) = 1$ and $d \geq 1$ be the degree of alignment of $\mathbf{v}$ with W, then*

$$\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] = \Theta(t^{d-0.5}).$$

Proof. For notational convenience, we define $\boldsymbol{v}_k \overset{\text{def}}{=} \mathbf{v}\epsilon_k$. Before we show either bound, we first note that using the independence of ϵ_t and $\epsilon_{t'}$ for $t \neq t'$, we can write

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{x}_t\|^2\right] &= \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k \boldsymbol{v}_t\right\|^2\right] \\
&= \mathbb{E}\left[\left(\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k}\right)^\intercal \left(\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k}\right)\right] \\
&= \mathbb{E}\left[\sum_{k=0}^{t-1}\sum_{k'=0}^{t-1} \left(W^k \boldsymbol{v}_{t-k}\right)^\intercal \left(W^{k'} \boldsymbol{v}_{t-k'}\right)\right] \\
&= \sum_{k=0}^{t-1}\sum_{k'=0}^{t-1} \mathbb{E}\left[\left(W^k \boldsymbol{v}_{t-k}\right)^\intercal \left(W^{k'} \boldsymbol{v}_{t-k'}\right)\right] \\
&= \sum_{k=0}^{t-1} \mathbb{E}\left[\left(W^k \boldsymbol{v}_{t-k}\right)^\intercal \left(W^k \boldsymbol{v}_{t-k}\right)\right] \\
&= \sum_{k=0}^{t-1} \mathbb{E}\left[\left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] \\
&= \mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right]. \qquad (6.9)
\end{aligned}$$

Upper Bound: First we note that by Jensen's inequality

$$\mathbb{E}\left[\|\boldsymbol{x}_t\|^2\right] \geq \left(\mathbb{E}\left[\|\boldsymbol{x}_t\|\right]\right)^2.$$

Taking square roots of both sides gives

$$\sqrt{\mathbb{E}\left[\|\boldsymbol{x}_t\|^2\right]} \geq \sqrt{\left(\mathbb{E}\left[\|\boldsymbol{x}_t\|\right]\right)^2} = \mathbb{E}\left[\|\boldsymbol{x}_t\|\right]$$

Using this, along with Equation 6.9 we have

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &\leq \sqrt{\mathbb{E}\left[\|\boldsymbol{x}_t\|^2\right]} \\
&= \sqrt{\mathbb{E}\left[\sum_{k=0}^{t-1} \|W^k \boldsymbol{v}_{t-k}\|^2\right]},
\end{aligned}$$

which means by Lemma 6.4.1, we have the existence of $C_3 > 0$ and t_0 such that for all $t > t_0$,

$$\begin{aligned}\mathbb{E}\left[\|\mathbf{x}_t\|\right] &< \sqrt{(C_3 \cdot t^{2\cdot d-1})} \\ &= \sqrt{C_3} \cdot t^{d-0.5}\end{aligned}$$

We therefore have that, for all large enough t,

$$\mathbb{E}\left[\|\mathbf{x}_t\|\right] < C_1 \cdot t^{d-0.5}. \tag{6.10}$$

where $C_1 \overset{\text{def}}{=} \sqrt{C_3}$.

Lower Bound: For each $i \in \mathbb{N}$, let r_i be a Rademacher random variable, that is, a random variable with $\mathbb{P}(r_i = 1) = \mathbb{P}(r_i = -1) = 0.5$. For all t, by the inequality in Theorem A.2.5, we have

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &= \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k}\right\|\right] \\ &\geq \frac{1}{2} \cdot \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k} r_{t-k}\right\|\right],\end{aligned}$$

We are now able to make use of the Khinchin-Kahane inequality (as defined in Theorem A.2.6) to get

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &\geq \frac{1}{2} \cdot \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k} r_{t-k}\right\|\right] \\ &\geq \frac{1}{2} \cdot \frac{1}{C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k} r_{t-k}\right\|^2\right]} \\ &= \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\left(\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k} r_{t-k}\right)^{\intercal}\left(\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k} r_{t-k}\right)\right]} \\ &= \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\sum_{k=0}^{t-1} (W^k \boldsymbol{v}_{t-k} r_{t-k})^{\intercal} (W^k \boldsymbol{v}_{t-k} r_{t-k})\right]} \\ &= \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\sum_{k=0}^{t-1} (W^k \boldsymbol{v}_{t-k})^{\intercal} (W^k \boldsymbol{v}_{t-k})\right]}\end{aligned}$$

$$= \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\sum_{k=0}^{t-1} \|W^k \boldsymbol{v}_{t-k}\|^2\right]}.$$

Now we apply Lemma 6.4.1 in order to get

$$\begin{aligned} \mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &\geq \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\sum_{k=0}^{t-1} \|W^k \boldsymbol{v}_{t-k}\|^2\right]} \\ &> \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{C_4 \cdot t^{2 \cdot d - 1}} \\ &= C_2 \cdot t^{d-0.5}, \end{aligned}$$

where $C_2 \overset{\text{def}}{=} \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{C_4}$. □

From here, we can expand to the multi-dimensional case without too much difficulty.

Corollary 6.4.1. *Let $\{\boldsymbol{u}_t\}_{t=1}^{\infty}$ be a sequence of random vectors in $\mathbb{R}^D$, with the entries of all vectors being independent with zero-mean and a common non-zero finite variance. For the ESN (W, V) driven by that sequence, if $\rho(W) = 1$ and there exists at least one column of V aligned with W with degree at least one, then*

$$\mathbb{E}\,\|\boldsymbol{x}_t\| = \Theta(t^{d-0.5}).$$

Proof. First we note that

$$\begin{aligned} \boldsymbol{x}_t &= \sum_{k=0}^{t-1} W^k V \boldsymbol{u}_{t-k} \\ &= \sum_{i=1}^{D} \sum_{k=0}^{t-1} W^k V_{\cdot,i} (\boldsymbol{u}_{t-k})_i, \end{aligned}$$

where $V_{\cdot,i}$ is the ith column of V. Define

$$\boldsymbol{x}_t^{(i)} \overset{\text{def}}{=} \sum_{k=0}^{t-1} W^k V_{\cdot,i} (\boldsymbol{u}_{t-k})_i,$$

We can write $\boldsymbol{x}_t = \sum_{i=1}^{D} \boldsymbol{x}_t^{(i)}$, giving that $\boldsymbol{x}_t$ is the sum of independent random vectors. The upper bound can easily be obtained by splitting the norm of this sum into a sum of norms using the triangle inequality, then applying Theorem 6.4.1. That is to say, we have

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &= \mathbb{E}\left[\left\|\sum_{i=1}^{D} \boldsymbol{x}_t^{(i)}\right\|\right] \\
&\leq \mathbb{E}\left[\sum_{i=1}^{D} \left\|\boldsymbol{x}_t^{(i)}\right\|\right] \\
&< D \cdot C_2 \cdot t^{d-0.5}
\end{aligned}$$

For the lower bound, we use the independence of each $\boldsymbol{x}_t^{(i)}$, Theorem A.2.5 and the Khinchin-Kahane inequality to get

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &= \mathbb{E}\left[\left\|\sum_{i=1}^{D} \boldsymbol{x}_t^{(i)}\right\|\right] \\
&\geq \frac{1}{2} \cdot \mathbb{E}\left[\left\|\sum_{i=1}^{D} r_i \boldsymbol{x}_t^{(i)}\right\|\right] \\
&\geq \frac{1}{2} \cdot \frac{1}{C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\left\|\sum_{i=1}^{D} r_i \boldsymbol{x}_t^{(i)}\right\|^2\right]} \\
&= \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\sum_{i=1}^{D} \mathbb{E}\left[\left\|r_i \boldsymbol{x}_t^{(i)}\right\|^2\right]} \\
&= \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\sum_{i=1}^{D} \mathbb{E}\left[\left\|\boldsymbol{x}_t^{(i)}\right\|^2\right]} \\
&= \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\left\|\sum_{i=1}^{D} \boldsymbol{x}_t^{(i)}\right\|^2\right]} \\
&\geq \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\left\|\boldsymbol{x}_t^{(j)}\right\|^2\right]} \\
&> \sqrt{C_1' \cdot t^{2 \cdot t - 1}} > C_1 \cdot t^{d-0.5},
\end{aligned}$$

where $V_{\cdot j}$ is one of the columns of V which has the maximum degree of alignment d. $\square$

We are not usually interested in the behaviour of a network subjected to purely random input. We do, however, frequently consider data drawn from noisy sources, particularly data which has been perturbed by Gaussian random noise. This is the case that we deal with in the following corollary.

Corollary 6.4.2. *Let (W, V) be an ESN satisfying $\rho(W) = 1$, with $V \in \mathbb{R}^D$ and let $d \geq 1$ be the maximum degree of alignment of W with any of the columns of V. Let $\mathbf{x}_t$ be the state of (W, V) at time t driven by the input sequence $\{\boldsymbol{u}_t\}_{t=1}^{\infty}$, where for each t, $\boldsymbol{u}_t = \mathbf{u}_t + \boldsymbol{\epsilon}_t$, where each $\mathbf{u}_t$ is in $\mathbb{R}^D$ and each entry of each $\boldsymbol{\epsilon}_t$ is an i.i.d. random variable, and all entries of all $\boldsymbol{\epsilon}_t$ have the same non-zero variance. There exist $C_1, C_2 > 0$ and t_0 such that for all $t > t_0$,*

$$C_1 \cdot t^{d-0.5} < \mathbb{E}\left[\|\boldsymbol{x}_t\|\right] < \left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{u}_t\right\| + C_2 \cdot t^{d-0.5}.$$

Proof. For the upper bound, we can simply use the triangle inequality to get:

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &= \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{u}_{t-k}\right\|\right] \\
&= \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \left(\mathbf{u}_{t-k} + \boldsymbol{\epsilon}_{t-k}\right)\right\|\right] \\
&= \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} + \sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right] \\
&\leq \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k}\right\| + \left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right]
\end{aligned}$$

Combining this with Theorem 6.4.1 on the above gives for sufficiently large t

$$\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] < \left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{u}_t\right\| + C_2 \cdot t^{d-0.5}.$$

For the lower bound, we use Theorem 6.4.1 to say that for some constants C_3 and all $t_0 > t$

$$C_3 \cdot t^{d-0.5} < \left\|\sum_{t=0}^{k-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|. \qquad (6.11)$$

Now choose $C_4 < C_3$ and define $\mathbf{x}'_t \overset{\text{def}}{=} \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k}$. For all $t > t_0$, if $\|\mathbf{x}'_t\| > C_4 \cdot t^{d-0.5}$, the convexity of the norm along with Jensen's inequality gives

$$\begin{aligned}
\mathbb{E}\left[\|\mathbf{x}_t\|\right] &= \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} + \sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right] \\
&\geq \left\|\sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} + \underbrace{\mathbb{E}\left[\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right]}_{=0}\right\| \\
&= \left\|\sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k}\right\| \\
&> C_4 \cdot t^{d-0.5}
\end{aligned}$$

Otherwise, we have $\|\mathbf{x}'_t\| \leq C_4 \cdot t^{d-0.5}$ and by applying the reverse triangle inequality, then Jensen's inequality we get for such t,

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &\geq \mathbb{E}\left[\left|\|\mathbf{x}'_t\| - \left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right|\right] \\
&\geq \left|\mathbb{E}\left[\|\mathbf{x}'_t\| - \left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right]\right| \\
&= \left|\|\mathbf{x}'_t\| - \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right]\right| \\
&= \left|\mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right] - \|\mathbf{x}'_t\|\right|.
\end{aligned}$$

We can now use Equation 6.11 to get

$$\begin{aligned}
\left|\mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right] - \|\mathbf{x}'_t\|\right| &\geq \left|\underbrace{\mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right]}_{>C_3 \cdot t^{d-0.5}} - \underbrace{\|\mathbf{x}'_t\|}_{\leq C_4 \cdot t^{d-0.5}}\right| \\
&> |(C_3 - C_4)| \cdot t^{d-0.5}
\end{aligned}$$

Combining the two cases, we can complete the proof by setting $C_1 \overset{\text{def}}{=} \min(C_4, C_3 -$

C_4). □

Note the dependence of the upper bound on the norm of the deterministic part. In general, we do not have assurances that the deterministic part grows at the same asymptotic rate, even if we make the assumption that all inputs are form a compact subset of the input space. Consider for example the case where $W = J_d(1)$ [4] for some d and $(\mathbf{v})_i = \delta_{id}$ for $1 \leq i \leq d$ and the input sequence $u_t = 1$ for all u_t. This would give that for each k,

$$\left(W^k \mathbf{v}\right)_1 = \frac{k^{d-1}}{(d-1)!} + o(k^{d-1})$$

Since this component of $W^k \mathbf{v}$ is positive for all k, we have

$$\begin{aligned}(\mathbf{x}'_t)_i &= \sum_{k=0}^{t-1} \frac{k^{d-1}}{(d-1)!} + o(k^{d-1}) \\ &= C \cdot t^d + o(t^d),\end{aligned}$$

for some positive constant C. This gives that the norm $\|\mathbf{x}'_t\|$ grows at a rate at least commensurate with t^d. Though we omit proof, it is possible to use the results of [Lea94] to show that for a compact input space, this is the upper bound on the rate of growth.

6.5 The Explosive Case: $\rho(W) > 1$

The final case we consider is $\rho(W) > 1$. In this section, we once again wish to find the rate of growth of the expectation of the norm of the hidden state. As one might expect, the rate of growth is exponential and determined by the spectral radius. Calculating this uses a similar approach to that in the previous section. We begin with a lemma analogous to Lemma 6.4.1. Proof of this lemma is provided in Appendix D.2.

Lemma 6.5.1. *Let W be a matrix such that $\rho(W) > 1$, and $\mathbf{v}$ be a vector which is aligned with W with degree $d \geq 1$. Define $\boldsymbol{v}_k = \mathbf{v}\epsilon_k$, where each ϵ_k is a random variable with zero mean and the same non-zero finite variance. For any $\varepsilon > 0$,*

[4] We remind the reader that $J_d(1)$ is the $d \times d$ Jordan block with 1's on the diagonal, as defined in AppendixA.1

there exist constants $t_0 > 0$, $C_1 > 0$ and $C_2 > 0$ such that for all $t \geq t_0$

$$C_1 \cdot \rho(W)^{2\cdot k} < \mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] < C_2 \cdot (\rho(W) + \varepsilon)^{2\cdot k}.$$

In the same way that Lemma 6.4.1 is used in the previous section, we are able to leverage Lemma 6.5.1 in order to derive inequalities for the asymptotic behaviour of our network.

Theorem 6.5.1. *Let $\boldsymbol{x}_t$ be the state of an ESN $(W, \mathbf{v})$ driven by a sequence of random variables $\{\epsilon_t\}_{t=1}^{\infty}$, each with zero mean and the same fixed variance. If $\rho(W) > 1$ and $\mathbf{v}$ is aligned to W with degree $d \geq 1$, then for all $\varepsilon > 0$ there exist constants $C_1 > 0, C_2 > 0, t_0 > 0$ such that for all $t > t_0$,*

$$C_1 \cdot \rho(W)^t < \mathbb{E}\left[\|\boldsymbol{x}_t\|\right] < C_2 \cdot (\rho(W) + \varepsilon)^t.$$

Proof. We again define $\boldsymbol{v}_{t-k} \overset{\text{def}}{=} \mathbf{v}\epsilon_{t-k}$. For the upper bound we use Jensen's inequality and Equation 6.9 to get

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &\leq \sqrt{\mathbb{E}\left[\|\boldsymbol{x}_t\|^2\right]} \\ &= \sqrt{\sum_{k=0}^{t-1} \mathbb{E}\left[\|W^k \boldsymbol{v}_{t-k}\|^2\right]}.\end{aligned}$$

Using the second inequality in Lemma 6.5.1, we get for large enough t and some constant $C_3 > 0$

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &< \sqrt{C_3 \cdot (\rho(W) + \varepsilon)^{2\cdot t}} \\ &= C_2 \cdot (\rho(W) + \varepsilon)^t\end{aligned}$$

Where $C_2 \overset{\text{def}}{=} \sqrt{C_3}$. For the lower bound, we use Theorem A.2.5 and the Khinchin-Kahane inequality to get for Rademacher random variables $r_1, \dots r_t$,

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &= \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k \boldsymbol{v}_{t-k}\right\|\right] \\ &\geq \frac{1}{2} \cdot \mathbb{E}\left[\left\|\sum_{k=0}^{t-1} r_{t-k} W^k \boldsymbol{v}_{t-k}\right\|\right]\end{aligned}$$

$$\begin{aligned}
&\geq \frac{1}{2 \cdot C_{1,2}} \sqrt{\mathbb{E}\left[\left\|\sum_{k=0}^{t-1} r_i W^k \boldsymbol{v}_{t-k}\right\|\right]^2} \\
&= \frac{1}{2 \cdot C_{1,2}} \sqrt{\mathbb{E}\left[\sum_{k=0}^{t-1} \|r_i W^k \boldsymbol{v}_{t-k}\|\right]^2} \\
&= \frac{1}{2 \cdot C_{1,2}} \sqrt{\mathbb{E}\left[\sum_{k=0}^{t-1} \|W^k \boldsymbol{v}_{t-k}\|\right]^2}.
\end{aligned}$$

We are now able to apply the lower bound in Lemma 6.5.1 to get for sufficiently large t,

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &> \frac{1}{2 \cdot C_{1,2}} \sqrt{C_1' \cdot \rho(W)^{2 \cdot t}} \\
&= C_1 \cdot \rho(W)^t
\end{aligned}$$

where $C_1 \overset{\text{def}}{=} \frac{\sqrt{C_1'}}{2 \cdot C_{1,2}}$. □

As with the unstable case, this result can be extended in a fairly natural manner to the case of multi-dimensional input.

Corollary 6.5.1. *Let $\{\boldsymbol{u}_t\}_{t=1}^{\infty}$ be a sequence of random vectors in $\mathbb{R}^D$, with the entries of all vectors being zero-mean and a common non-zero finite variance. For the ESN (W, V) driven by that sequence, if $\rho(W) > 1$ and there exists at least one column of V aligned with W with degree at least one, then for all $\varepsilon > 0$ there exist $C_1 > 0$, $C_2 > 0$, $t_0 > 0$ such that for all $t > t_0$,*

$$C_1 \cdot \rho(W)^t < \mathbb{E}\left[\|\mathbf{x}_t\|\right] < C_2 \cdot \rho(W + \varepsilon)^t.$$

The proof of the above corollary is identical in structure to Corollary 6.4.1, but makes use of Theorem 6.5.1 and Lemma 6.5.1 instead of Theorem 6.4.1 and Lemma 6.4.1, respectively. As such, we omit the full proof from the main text, and instead include it in Appendix D.2.

Similarly, we can can adapt Corollary 6.4.2 to this case as follows

Corollary 6.5.2. *Let (W, V) be an ESN satisfying $\rho(W) > 1$. Let $d \geq 1$ be the greatest degree of alignment of columns of V with W. Let $\mathbf{x}_t$ be the state of the ESN (W, V) driven by the input sequence $\{\boldsymbol{u}_t\}_{t=1}^{\infty}$ where for each t, $\boldsymbol{u}_t = \mathbf{u}_t + \boldsymbol{\epsilon}_t$*

with each $\mathbf{u}_t$ *as some vector in* $\mathbb{R}^D$ *and each* $\boldsymbol{\epsilon}_t$ *as a vector of the same size whose entries are zero-mean i.i.d. random variables, with the entries of all* $\boldsymbol{\epsilon}_t$ *having the same non-zero variance. For all* $\varepsilon > 0$*, there exist* $C_1, C_2, t_0 > 0$ *such that for all* $t > t_0$*,*

$$C_1 \cdot \rho(W)^t < \mathbb{E}\left[\|\boldsymbol{x}_t\|\right] < \left\| \sum_{k=0}^{t-1} W^k V \mathbf{u}_t \right\| + C_2 \cdot (\rho(W) + \varepsilon)^t.$$

Again, the structure of the proof of this corollary is identical to the equivalent result in the unstable case (in this case Corollary 6.4.2). As such, we once again relegate the full proof to Appendix D.2.

6.6 Convergence of Hidden States

The echo state property is of interest because it guarantees that the initial state of the network is inconsequential to the network's long-term behaviour. However, in order for the network's hidden state to be a useful representation of the input, it is also necessary for the trajectory of the hidden state through the state space to be defined by the input sequence. The following theorem gives some aymptotic guarantees on the network behaviour in this regard.

Theorem 6.6.1. *Let* $\mathbf{x}_t$ *and* $\mathbf{y}_t$ *be the trajectories of the hidden states of two linear ESNs, both with recurrent weight matrix* W *satisfying* $\rho(W) < 1$ *and input weight matrix* V*, and driven by input sequences* $\{\mathbf{u}_t\}_{t=1}^{\infty}$ *and* $\{\mathbf{v}_t\}_{t=1}^{\infty}$*, respectively. Then* $\|\mathbf{x}_t - \mathbf{y}_t\| \to 0$ *if and only if* $\|V(\mathbf{u}_t - \mathbf{v}_t)\| \to 0$*.*

Proof. $(\Rightarrow)$ We define $\mathbf{z}_t \overset{\text{def}}{=} \mathbf{x}_t - \mathbf{y}_t$ and note that

$$\begin{aligned}\mathbf{z}_t &= \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} - \sum_{k=0}^{t-1} W^k V \mathbf{v}_{t-k} \\ &= \sum_{k=0}^{t-1} W^k V \left(\mathbf{u}_{t-k} - \mathbf{v}_{t-k}\right) \\ &= W\mathbf{z}_{t-1} + V(\mathbf{u}_t - \mathbf{v}_t). \qquad (6.12)\end{aligned}$$

Now, by assumption $\|\mathbf{z}_t\| \to 0$, meaning we also get $\|W\mathbf{z}_t\| \to 0$ (since $\|W\mathbf{z}_t\| \leq$

$\|W\| \, \|\mathbf{z}_t\|)$. By rearrangement of Equation 6.12, we have

$$\begin{aligned} \|V(\mathbf{u}_t - \mathbf{v}_t)\| &= \|\mathbf{z}_{t+1} - W\mathbf{z}_t\| \\ &\leq \|\mathbf{z}_{t+1}\| + \|W\mathbf{z}_t\| , \end{aligned}$$

and since both terms on the right hand side go to zero as $t \to \infty$, $\|V(\mathbf{u}_t - \mathbf{v}_t)\|$ must also go to zero.

($\Leftarrow$) For the converse, we note that by unrolling $V(\mathbf{u}_t - \mathbf{v}_t) = \mathbf{z}_{t+1} - W\mathbf{z}_t$, we can obtain the more general expression

$$\mathbf{z}_t - W^k \mathbf{z}_{t-k} = \sum_{i=0}^{k-1} W^i V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}).$$

Rearranging and considering the norms of each side, we can use the triangle inequality to give

$$\begin{aligned} \|\mathbf{z}_t\| &= \left\| W^k \mathbf{z}_{t-k} + \sum_{i=0}^{k-1} W^i V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}) \right\| \\ &\leq \left\| W^k \mathbf{z}_{t-k} \right\| + \left\| \sum_{i=0}^{k-1} W^i V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}) \right\| \\ &\leq \left\| W^k \right\| \cdot \|\mathbf{z}_{t-k}\| + \left\| \sum_{i=0}^{k-1} W^i V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}) \right\| . \end{aligned} \tag{6.13}$$

From Theorem 6.3.1, we know that for $\rho(W) < 1$, the norm of the hidden states $\mathbf{x}_t$ and $\mathbf{y}_t$ are bounded by some constant C_{state}, so we can choose a constant C such that $C > 2 \cdot C_{state}$. This gives that the norm of $\mathbf{z}_t$ is bounded above by C. By Lemma A.3.1, for any $\varepsilon > 0$, we can also choose $k_0 \in \mathbb{N}$ such that for all $k \geq k_0$, $\left\| W^k \right\| < \varepsilon/(2 \cdot C)$ Setting $k = k_0$ in Equation 6.13, we get that the first term in the right hand side of the final inequality is less than. $\varepsilon/(2 \cdot C) \cdot C = \varepsilon/2$. We now want to show that for any fixed k the second term on the right hand side tends to zero as $t \to \infty$. Using the triangle inequality again gives

$$\left\| \sum_{i=0}^{k_0-1} W^i V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}) \right\| \leq \sum_{i=0}^{k_0-1} \left\| W^i \right\| \cdot \left\| V\left(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}\right) \right\| .$$

Since $\rho(W) < 1$, by Lemma A.2.1 there exists a (sub-multiplicative) norm $|||\cdot|||$ such that $|||W||| < 1$ and constant $C_1 > 0$ such that $\|X\| \leq C_1 \cdot |||X|||$ for all $X \in \mathbb{R}^m$, therefore,

$$\sum_{i=0}^{k_0-1} \|W^i\| \leq \sum_{i=0}^{\infty} \|W^i\| \leq C_1 \sum_{i=0}^{\infty} |||W^i||| \leq C_1 \sum_{i=0}^{\infty} |||W|||^i = \frac{C_1}{1 - |||W|||}.$$

Since $\|V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i})\| \to 0$, we can choose t_0 such that for any ε, $\|V(\mathbf{u}_t - \mathbf{v}_t)\| < {}^{(1-|||W|||)\cdot\varepsilon}/_{(2\cdot C_1)}$ for all $t > t_0$. giving

$$\left\| \sum_{i=0}^{k_0-1} W^i V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}) \right\| < \frac{C_1}{1 - |||W|||} \cdot \frac{(1 - |||W|||) \cdot \varepsilon}{2 \cdot C_1} = \frac{\varepsilon}{2}$$

for all $t > t_0 + k_0 - 1$. Plugging this back into Equation 6.13 gives us:

$$\begin{aligned} \|\mathbf{z}_t\| &\leq \left\|W^{k_0}\right\| \cdot \|\mathbf{z}_{t-k}\| + \left\| \sum_{i=0}^{k_0-1} W^i V(\mathbf{u}_{t-i} - \mathbf{v}_{t-i}) \right\| \\ &< \frac{\varepsilon}{2} + \frac{\varepsilon}{2} \\ &= \varepsilon. \end{aligned}$$

So for any ε, we can find t_0, such that $\|\mathbf{z}_t\| < \varepsilon$ for all $t > t_0$, and consequently $\lim_{t\to\infty} \|\mathbf{z}_t\| = 0$. □

For the case $\rho(W) \geq 1$, the forward direction of this proof still holds. The backwards direction, however, does not. As a simple example of this, consider $\mathbf{u}_1 = a$ for $a \neq 0$ and $\mathbf{u}_i = \mathbf{v}_j = 0$ for all $i > 1$ and all $j > 0$. Choose $\mathbf{p}$ to be an eigenvector of W associated with an eigenvalue with the same magnitude as the spectral radius of W and choose V to be a vector such that $\mathbf{p}^\intercal V \neq 0$. Clearly, for these inputs, $\|V(\mathbf{u}_t - \mathbf{v}_t)\| \to 0$. Define v to be the component of V in the direction of $\mathbf{p}$. We consider the norm $\|\cdot\|_P$ and use the equivalence of norms, to get that for constant $C_2 > 0$

$$\begin{aligned} \mathbf{z}_t &= \left\|W^{t-1} V \mathbf{u}_1\right\| \\ &\geq C_2 \cdot \left\|W^{t-1} V \mathbf{u}_1\right\|_P \\ &\geq C_2 \cdot |\rho(W)|^{t-1} \cdot |v \cdot a| \\ &\geq C_2 \cdot |\rho(W)| \cdot |v \cdot a|. \end{aligned}$$

The last expression is greater than zero, and not dependent on t, so $\mathbf{z}_t$ cannot converge to zero.

6.7 Conclusion

In this chapter we have clarified the relationship between the spectral radius of the recurrent matrix, $\rho(W)$, and the asymptotic behaviour of the network. For the case of $\rho(W) < 1$, we have provided proof of the boundedness of the hidden norm of the hidden state, incorporating standard results from linear systems theory previously absent from the echo state network literature; we have also shown the sufficiency of $\rho(W) < 1$ for the echo state property, and shown that sub-Gaussian input is sufficient to guarantee sub-exponential tails in the distribution of the hidden state of the network. For the case $\rho(W) \geq 1$, we have described the rate of growth of the expectation of the norm of the hidden state, showing that in the case $\rho(W) = 1$, the rate of growth is polynomial, with the degree determined by the structure of W and the direction of $\mathbf{v}$ or V. For $\rho(W) > 1$, we have shown that the rate of growth is exponential, again conditioned on the direction of $\mathbf{v}$ or V. Finally, we have provided insight into how the asymptotic behaviour of the network is dependent upon the asymptotic behaviour of the input driving it.

In terms of practical considerations for ESNs, the key insight from this chapter is not an especially novel one: in linear networks, one should generally ensure that $\rho(W) < 1$. However, we have shown *why* this advice is so important in linear networks, giving insight into the desirable properties that we get when the condition is satisfied, and showing how things go wrong when it is not. This emphasizes the contrast between the linear and non-linear cases: when a saturating non-linearity is applied, it is useful to explore the possibility of larger radii [LJ09], using the boundedness of activation function to ensure stability even though the echo state property does not hold. Additionally, we have shown that spectral radius less than one is sufficient condition for the echo state property in the linear case, again in contrast to the non-linear case, where there is a significant gap between known necessary conditions and sufficient ones [YJK12].

These considerations about the behaviour of linear echo state networks are increasingly important in practice, as researchers experiment with methods of increasing network memory that include constructing networks with both linear and non-linear components [IY17; GGM18]. In theoretical work, the simplification from non-linear to linear dynamics allows for analysis of the effects of different reservoir structures [WLS04; RT11a; Rod12].

Finally, we once again emphasise the generality of these results: they hold not just for ESNs, but for any linear input-driven system of the forms given in Equation 6.1 and Equation 6.2. And while we can view these results as an important part of completing our understanding of linear ESNs, further applications of this work are as likely to be in domains which are outside the scope of this thesis as they are to be within ESNs, or even machine learning more broadly.

Chapter 7

Conclusion

In this thesis, our goal was to contribute knowledge towards an answer to the question: “How does the structure of an Echo State Network affect its behaviour?”. In this chapter, we examine the contributions made in the thesis, and discuss to what extent they achieve this goal. We also discuss how these contributions fit into and expand upon the existing body of knowledge on the subject. Finally, we close the chapter, and the thesis, by proposing future directions of research which build upon the work which we have presented.

7.1 Summary of Research

In this section, we summarise the contributions made in each of the three research chapters of this thesis, as well as briefly discuss how the work contributes to the literature.

7.1.1 Memory in Linear Networks

In Chapter 4, we examined how the memory capacity of a network could be inferred from the structure of its weights. Building on [Jae02b] and results from control theory, we constructed an explicit expression for the memory capacity of a network based on the Jordan decomposition of its recurrent weight matrix and the direction of its input weight vector. We showed how these results could be used to give explicit memory capacity measurements for existing network designs, including both deterministic and random network initialisation schemes. By examining random initialisations we found, perhaps surprisingly, that almost all possible reservoirs have the maximum possible memory capacity. We also extended our analysis to deep linear networks, and by deriving an expression that allows us to construct a single-layer network with the same dynamics as an arbitrary deep network, we showed almost all deep linear networks achieve this maximum memory capacity, even if we impose the restriction that the same weight matrices are repeated by each layer.

This is an important contribution to the literature, as it gives a new perspective on memory capacity and a novel tool for analytically determining the memory capacity of a given network. Additionally, we provide several demonstrations of the power of this tool in determining the memory capacity of existing reservoir structures. While we showed that achieving the maximum possible memory capacity is easy, in the sense that randomly generated networks will achieve it with probability one, we also noted that this was with the assumption of infinite precision computation. In practice, with finite precision hardware, the empirical memory capacity of networks can vary greatly from the theoretical upper bound (as we saw, for example, in Chapter 5). This is an important caveat to the work, but our contribution still provides a useful necessary condition that a reservoir structure must satisfy in order to achieve the maximum memory capacity, even if in practice it is not a sufficient condition.

7.1.2 Depth in Recurrent Networks

In Chapter 5, we examined the previously reported, but largely unexplained, phenomenon in which different layers of deep recurrent networks exhibit different behaviours in how their response to inputs varies over time. We refer to this as the ‘different time-scales phenomenon’. Through a series of careful experiments, we examined two particular manifestations of the phenomenon: sensitivity to perturbations of input and layer-wise memory capacity. In both cases, we demonstrated that the effect was present in linear networks, and explored the sensitivity of the phenomenon to network hyperparameters.

Though the different time-scales phenomenon has been observed previously, we are able to offer several novel insights into the nature and cause of the phenomenon. In the linear case, we establish evidence of a simple relationship between layer depth and the effects of input perturbation, as well as showing how this phenomenon is affected by the spectral radius of the recurrent weights. In examining the memory capacity of these systems, we are able to utilise results from Chapter 4 in order to provide insights into the cause of the difference in memory between layers. Additionally, in the non-linear case, we highlight the importance of considering the scaling between layers in controlling the properties of the network.

7.1.3 The Asymptotic Behaviour of Linear Networks

In Chapter 6, we examined the role of the recurrent weight matrix W in determining the asymptotic behaviour of the network on arbitrarily long input sequences. In particular, we clarified the relationship between the spectral radius of W and two important notions of stability: external stability and the echo state property. We showed that both of these forms of stability are achieved by the same condition in linear networks, namely $\rho(W) < 1$. For such networks, we showed the effects of sub-Gaussian noise on the network state, showing that if the tails of the input distribution are light, then so will be the tails of the distribution of the components of the hidden state. For unstable networks, we showed that the expected rate of growth of the norm is polynomial in the case $\rho(W) = 1$, and exponential in the case $\rho(W) > 1$. We also showed a complementary property to the echo state property holds in linear networks with $\rho(W) < 1$, namely, the distance between network states of identical networks which are fed distinct sequences converges if and only if those sequences converge.

The role of this chapter in the literature is primarily in terms of clarifying and completing our understanding of the behaviour of ESNs, particularly in terms of stability. By showing that the ESP holds for all W satisfying $\rho(W) < 1$ in the linear case, we highlight an important distinction between the linear and non-linear cases. Additionally, by examining unstable network configurations, we provide a more complete picture of the dynamics of linear input-driven systems. While we do not claim any direct applications of these result, they improve our understanding of the behaviour of an oft-neglected class of such systems.

7.2 Outlook and Future Work

In this section, we discuss the directions of future research motivated by work in this thesis.

7.2.1 Memory in Linear Networks

An important limitation of the work in Chapter 4 is our assumption that we are able to perform computation with infinite precision. Since this is not the case in practice, an important direction of future research is the exploration of the effects of finite precision computation on the memory capacity of networks with randomly sampled weights. Particularly, developing the tools to be able to reason about the expected memory capacity of a random network given known constraints on the network's precision would be useful for predicting network behaviour without relying on empirical study.

Another direction of research was suggested within the chapter itself, namely exploring the conditions for maximum memory capacity in deterministic reservoir constructions. In particular, it would be useful to be able to generate explicit condition or simple heuristics which guarantee the maximum possible memory capacity for ESNs for all deterministic structures discussed in Section 3.4.2. There are also other interesting reservoir structures to which our results for memory capacity may be applicable.

One of the key insights of the chapter was that we can draw parallels between the notion of memory capacity and the notion of controllability; this allowed us to drawn on existing results from control theory in our work. However, we have not fully explored the full suite of tools which this gives us access to, and it may be that further research could reveal more connections which may be useful in examining memory capacity in the non-linear case.

7.2.2 Depth in Recurrent Networks

In Chapter 5, our exploration of the different time-scales phenomenon was almost entirely empirical. This is an important first step in increasing our understanding of the phenomenon, and is useful when considering practical ways of leveraging this network behaviour, but it means there is still plenty more work to be done. The work presented in Chapter 5 offers multiple avenues of future research, both in further exploring the causes and consequences of the different time-scales phenomenon and in designing networks to more fully exploit our current understanding of it.

The work in the chapter does not attempt to satisfactorily explain *why* these phenomena occur. To this end, we suggest that it may be fruitful to take a more rigorous mathematical approach, rather than an empirical one (particularly in attempting to explain the differences in sensitivity to perturbation which we have observed between layers).

In terms of exploitation of the insights gained from this thesis, an important direction of research is in designing DeepESNs which to mitigate the numerical errors found in Section 5.4. By careful consideration of how spectral radius and layer size should vary between layers of the network, it may be possible to improve upon the usual network design, both in terms of layer-wise memory capacity and memory capacity of the network as a whole. Equally, it is important that we explore whether such networks are useful in practice, i.e., whether they offer improved performance for time-series prediction problems where the network is required to model long-term dependencies.

Additionally, it is our belief that these tools could have applications in recurrent networks more generally. In particular, a more rigorous understanding of these phenomena could lead to the development of more principled initialisation schemes for deep recurrent networks, in the same manner that Xavier/Glorot and Kaiming/He initialisation were developed for the case of feedforward networks. The aforementioned initialisation schemes are designed so that signals retain approximately the same magnitude when traversing through different layers of the network. Given the effects that we've observed when varying the feedfoward weights of a DeepESN, it is plausible that finding an initialisation scheme that accomplishes the same results for recurrent networks would prove similarly important to improving network training.

7.2.3 The Asymptotic Behaviour of Linear Networks

In Chapter 6, we examined the effects of network structure on the asymptotic behaviour of linear echo state networks. Though through our analysis we are able make stronger claims about the asymptotic behaviour than previously existed, further research could strengthen the claims further. For instance, while we make the claim about the rate of growth of the expectation of the norm of the hidden state, it may be desirable to strengthen the results by considering the concentration of norm of the state around its mean.

We make the claim that the work presented in Chapter 6 is applicable to a wider range of linear systems—and indeed there is nothing that particularly ties our description of the network to problems in machine learning. As such, it is possible that future work using the results in Chapter 6 could emerge from other domains. In particular, the results in the unstable and explosive cases may be applicable to methods of modelling of unstable systems which are beyond the scope of this thesis.

7.3 Closing Remarks

In this thesis, we have clarified several aspects of the relationship between the structure of an ESNs weights and the behaviour of the network. A common theme amongst the chapters of this thesis is that in the linear case, a lot of a network's behaviour can be inferred from just the Jordan decomposition of the recurrent weight matrix and the direction of the input vector $\mathbf{v}$. Indeed, when the recurrent weight matrix of W is simple, a lot of the behaviour can be inferred from the magnitude of the eigenvalues and whether $\mathbf{v}$ is orthogonal to any of the eigenvectors. Though we have focused primarily on the linear case, we have shown that even with this simplification our understanding remains incomplete, and there is plenty of fresh ground still to tread. In the non-linear case, there is even more that we still do not know, and we are likely to need new and more powerful techniques in order to advance our understanding.

Bibliography

[AR10] H Anton and C Rorres. *Elementary Linear Algebra: Applications Version*. John Wiley & Sons, 2010. ISBN: 9780470432051. URL: https://books.google.co.uk/books?id=1PJ-WHepeBsC.

[Bam18] Bassam Bamieh. *Discovering the Fourier Transform: A Tutorial on Circulant Matrices, Circular Convolution, and the DFT*. 2018. arXiv: 1805.05533. URL: http://arxiv.org/abs/1805.05533.

[BB18] Davide Bacciu and Andrea Bongiorno. "Concentric ESN: Assessing the Effect of Modularity in Cycle Reservoirs". In: *Proceedings of the International Joint Conference on Neural Networks* 2018-July (2018). DOI: 10.1109/IJCNN.2018.8489462. arXiv: 1805.09244. URL: https://arxiv.org/pdf/1805.09244.pdf.

[Ben09] Yoshua Bengio. "Learning Deep Architectures for AI". In: *Foundations and Trends in Machine Learning* 2.1 (2009), pp. 1–127. ISSN: 1935-8237. DOI: 10.1561/2200000006. arXiv: 0500581. URL: http://www.iro.umontreal.ca/bengioy.

[BF14] P. Barancok and I. Farkas. *Memory Capacity of Input-Driven Echo State Networks at the Edge of Chaos*. Ed. by Stefan Wermter et al. Vol. 8681. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 41–48. ISBN: 978-3-319-11178-0. DOI: 10.1007/978-3-319-11179-7-6. URL: http://cogsci.fmph.uniba.sk/{~}farkas/Papers/barancok-farkas.icann14.pdf.

[Bis06] Christopher M. Bishop. *Machine Learning and Pattern Recoginiton*. Springer, 2006. ISBN: 9780387310732.

[Boe+11] Joschka Boedecker et al. "Information processing in echo state networks at the edge of chaos." In: *Theory in biosciences = Theorie in den Biowissenschaften* (2011). ISSN: 1611-7530. DOI: 10.1007/s12064-

011-0146-8. URL: http://www.ncbi.nlm.nih.gov/pubmed/22147532.

[Bou19] Laurent Boué. *Real numbers, data science and chaos: How to fit any dataset with a single parameter*. 2019. arXiv: 1904.12320. URL: http://arxiv.org/abs/1904.12320.

[BP06] Štefan Babinec and Jií Pospíchal. "Merging Echo State and Feedforward Neural Networks for Time Series Forecasting". In: *ICANN 2006: Artificial Neural Networks ICANN 2006*. Springer, 2006, pp. 367–375. DOI: 10.1007/11840817_39. URL: http://link.springer.com/10.1007/11840817{_}39.

[BSF94] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult". In: *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166. ISSN: 10459227. DOI: 10.1109/72.279181. arXiv: arXiv:1211.5063v2. URL: http://deeplearning.cs.cmu.edu/pdfs/Bengio{_}94.pdf.

[BVS10] John Butcher, David Verstraeten, and Benjamin Schrauwen. "Extending reservoir computing with random static projections: a hybrid between extreme learning and RC". In: *18th European Symposium on Artificial Neural Networks (ESANN 2010* April (2010). URL: https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2010-99.pdf.

[BW89] Joel G. Broida and S Gill Williamson. *A Comprehensive Introduction to Linear Algebra*. Addison Wesley Longman Publishing Co, 1989. ISBN: 978-0201500653. URL: https://cseweb.ucsd.edu/{~}gill/CILASite/.

[BY06] Michael Buehner and Peter Young. "A tighter bound for the echo state property". In: *IEEE Transactions on Neural Networks* 17.3 (2006), pp. 820–824. ISSN: 10459227. DOI: 10.1109/TNN.2006.872357.

[BY86] Z. D. Bai and Y. Q. Yin. "Limiting behavior of the norm of products of random matrices and two problems of Geman-Hwang". In: *Probability Theory and Related Fields* 73.4 (1986), pp. 555–569. ISSN: 01788051. DOI: 10.1007/BF00324852.

[Cal+13] Ken Caluwaerts et al. "The spectral radius remains a valid indicator of the Echo state property for large reservoirs". In: *The 2013 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2013, pp. 1–6. ISBN: 978-1-4673-6129-3. DOI: 10.1109/IJCNN.2013.6706899. URL: http://ieeexplore.ieee.org/document/6706899/.

[Car+18] Zachariah Carmichael et al. *Mod-DeepESN: Modular Deep Echo State Network*. 2018. arXiv: 1808.00523. URL: https://arxiv.org/pdf/1808.00523.pdf.

[Cho+14] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: (2014). ISSN: 09205691. DOI: 10.3115/v1/D14-1179. arXiv: 1406.1078. URL: http://arxiv.org/abs/1406.1078.

[Cir+11] Dan C Ciresan et al. "Flexible, High Performance Convolutional Neural Networks for Image Classification". In: *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence Flexible* (2011), pp. 1237–1242. ISSN: 10450823. DOI: 10.5591/978-1-57735-516-8/IJCAI11-210. arXiv: arXiv:1011.1669v3.

[CPS18] Minmin Chen, Jeffrey Pennington, and Samuel S. Schoenholz. *Dynamical Isometry and a Mean Field Theory of RNNs: Gating Enables Signal Propagation in Recurrent Neural Networks*. 2018. arXiv: 1806.05394. URL: http://arxiv.org/abs/1806.05394.

[CSDS16] Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. *Capacity and Trainability in Recurrent Neural Networks*. 2016. arXiv: 1611.09913. URL: http://arxiv.org/abs/1611.09913.

[CST19] Mohammad Amin Chitsazan, M. Sami Fadali, and Andrzej M. Trzynadlowski. "Wind speed and wind direction forecasting using echo state network with nonlinear functions". In: *Renewable Energy* 131 (2019), pp. 879–889. ISSN: 18790682. DOI: 10.1016/j.renene.2018.07.060. URL: https://doi.org/10.1016/j.renene.2018.07.060.

[CT05] Richard Caron and Tim Traynor. *The zero set of a polynomial*. 2005. URL: http://www1.uwindsor.ca/math/sites/uwindsor.ca.math/files/05-03.pdf.

[Cyb89] G. Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of Control, Signals, and Systems* 2.4 (1989), pp. 303–314. ISSN: 0932-4194. DOI: 10.1007/BF02551274. URL: http://link.springer.com/10.1007/BF02551274.

[Dam+12] Joni Dambre et al. “Information Processing Capacity of Dynamical Systems”. In: *Scientific Reports* 2 (2012), p. 514. ISSN: 2045-2322. DOI: 10.1038/srep00514. URL: http://www.nature.com/doifinder/10.1038/srep00514.

[Doy92] K. Doya. “Bifurcations in the learning of recurrent neural networks”. In: *[Proceedings] 1992 IEEE International Symposium on Circuits and Systems*. Vol. 6. 4. IEEE, 1992, pp. 2777–2780. ISBN: 0-7803-0593-0. DOI: 10.1109/ISCAS.1992.230622. URL: http://ieeexplore.ieee.org/document/230622/.

[DS12] Ali Deihimi and Hemen Showkati. “Application of echo state networks in short-term electric load forecasting”. In: *Energy* 39.1 (2012), pp. 327–340. ISSN: 03605442. DOI: 10.1016/j.energy.2012.01.007. URL: http://dx.doi.org/10.1016/j.energy.2012.01.007.

[DZ07] Zhidong Deng and Yi Zhang. “Collective Behavior of a Small-World Recurrent Neural System With Scale-Free Distribution”. In: *IEEE Transactions on Neural Networks* 18.5 (2007), pp. 1364–1375. ISSN: 1045-9227. DOI: 10.1109/TNN.2007.894082. URL: http://ieeexplore.ieee.org/document/4298110/.

[EB96] Salah El Hihi and Yoshua Bengio. “Hierarchical recurrent neural networks for long-term dependencies”. In: *NIPS 8*. MIT Press, 1996. DOI: 10.1109/TDPVT.2004.1335423. URL: https://papers.nips.cc/paper/1102-hierarchical-recurrent-neural-networks-for-long-term-dependencies.pdf.

[Elm90] Jeffrey L Elman. “Finding Structure in Time”. In: *Cognitive Science* 14.2 (1990), pp. 179–211. ISSN: 03640213. URL: http://crl.ucsd.edu/{~}elman/Papers/fsit.pdf.

[FBG16] Igor Farkaš, Radomír Bosák, and Peter Gerge. “Computational analysis of memory capacity in echo state networks”. In: *Neural Networks* 83 (2016), pp. 109–120. ISSN: 08936080. DOI: 10.1016/j.neunet.

2016.07.012. URL: http://linkinghub.elsevier.com/retrieve/pii/S0893608016300946.

[FE05] Georg Fette and Julian Eggert. “Short term memory and pattern matching with simple echo state networks”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 3696 LNCS (2005), pp. 13–18. ISSN: 03029743. DOI: 10.1007/11550822_3.

[FG17] Igor Farkas and Peter Gergel. “Maximizing memory capacity of echo state networks with orthogonalized reservoirs”. In: *Proceedings of the International Joint Conference on Neural Networks* 2017-May.1 (2017), pp. 2437–2442. DOI: 10.1109/IJCNN.2017.7966152.

[Gal18] Claudio Gallicchio. *Short-term Memory of Deep RNN*. 2018. arXiv: 1802.00748. URL: http://arxiv.org/abs/1802.00748.

[GB10] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)* 9 (2010), pp. 249–256. ISSN: 15324435. DOI: 10.1.1.207.2059. URL: http://machinelearning.wustl.edu/mlpapers/paper{_}files/AISTATS2010{_}GlorotB10.pdf.

[Ge17] Stephen Cong Ge. “The Eigenvalue Spacing of IID Random Matrices and Related Least Singular Value Results”. PhD thesis. University of California, 2017.

[GGM18] Eleonora Di Gregorio, Claudio Gallicchio, and Alessio Micheli. “Combining Memory and Non-linearity in Echo State Networks”. In: *27th International Conference on Artificial Neural Networks* July (2018).

[GHS08] Surya Ganguli, Dongsung Huh, and Haim Sompolinsky. “Memory traces in dynamical systems.” In: *Proceedings of the National Academy of Sciences of the United States of America* 105.48 (2008), pp. 18970–5. ISSN: 1091-6490. DOI: 10.1073/pnas.0804451105. URL: http://www.ncbi.nlm.nih.gov/pubmed/19020074.

[GM16] Claudio Gallicchio and Alessio Micheli. “Deep Reservoir Computing: A Critical Analysis”. In: *European Symposium on Artificial Neural*

Networks, Computational Intelligence and Machine Learning April (2016), pp. 27–29.

[GM17] Claudio Gallicchio and Alessio Micheli. “Echo State Property of Deep Reservoir Computing Networks”. In: *Cognitive Computation* 9.3 (2017), pp. 337–350. ISSN: 18669964. DOI: `10.1007/s12559-017-9461-9`.

[GMP17] Claudio Gallicchio, Alessio Micheli, and Luca Pedrelli. “Deep reservoir computing: A critical experimental analysis”. In: *Neurocomputing* 268 (2017), pp. 87–99. ISSN: 18728286. DOI: `10.1016/j.neucom.2016.12.089`.

[GMP18] Claudio Gallicchio, Alessio Micheli, and Luca Pedrelli. “Design of deep echo state networks”. In: *Neural Networks* 108 (2018), pp. 33–47. ISSN: 18792782. DOI: `10.1016/j.neunet.2018.08.002`. URL: `https://doi.org/10.1016/j.neunet.2018.08.002`.

[GMP19] Claudio Gallicchio, Alessio Micheli, and Luca Pedrelli. “Hierarchical Temporal Representation in Linear Reservoir Computing”. In: *Neural Advances in Processing Nonlinear Dynamic Signals*. Ed. by Anna Esposito et al. Vol. 102. Smart Innovation, Systems and Technologies. Cham: Springer International Publishing, 2019, pp. 119–129. ISBN: 978-3-319-95097-6. DOI: `10.1007/978-3-319-95098-3_11`. arXiv: `1705.05782`. URL: `http://link.springer.com/10.1007/978-3-319-95098-3{\_}11`.

[GMS18] Claudio Gallicchio, Alessio Micheli, and Luca Silvestri. “Local Lyapunov exponents of deep echo state networks”. In: *Neurocomputing* 298 (2018), pp. 34–45. ISSN: 18728286. DOI: `10.1016/j.neucom.2017.11.073`. URL: `https://doi.org/10.1016/j.neucom.2017.11.073`.

[GO18] Lyudmila Grigoryeva and Juan Pablo Ortega. “Echo state networks are universal”. In: *Neural Networks* 108 (2018), pp. 495–508. ISSN: 18792782. DOI: `10.1016/j.neunet.2018.08.025`. arXiv: `1806.00797`.

[GO19] Lukas Gonon and Juan-Pablo Ortega. “Reservoir Computing Universality With Stochastic Inputs”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2019). ISSN: 2162-237X. DOI: `10.1109/tnnls.2019.2899649`. arXiv: `1807.02621`.

[Gra12] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. Vol. 385. Studies in Computational Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, p. 124. ISBN: 978-3-642-24796-5. DOI: `10.1007/978-3-642-24797-2`. arXiv: `arXiv:1308.0850v1`. URL: `https://www.cs.toronto.edu/{~}graves/preprint.pdf`.

[Gre+16] Klaus Greff et al. “LSTM: A Search Space Odyssey”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2016). ISSN: 21622388. DOI: `10.1109/TNNLS.2016.2582924`. arXiv: `1503.04069`.

[GV96] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)* John Hopkins University Press, 1996. ISBN: 0-8018-5413-X.

[Has86] Johan Hastad. “Almost Optimal Lower Bounds for Small Depth Circuits”. In: *Advances in Computing Research* (1986).

[Hau69] Malo L. J. Hautus. “Controllability and Observability Conditions of Linear Autonomous Systems”. In: *Proceedings of the Koninklijke Nederlandse Akademie van Wetenschappen, Series A — Mathematical Sciences* 72.5 (1969), p. 443.

[He+15] Kaiming He et al. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE International Conference on Computer Vision* 2015 Inter (2015), pp. 1026–1034. ISSN: 15505499. DOI: `10.1109/ICCV.2015.123`. arXiv: `1502.01852`.

[HJ85] Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge University Press, 1985. ISBN: 978-0-521-38632-6.

[Hor10] Kur Hornik. “Multilayer Feedforward Networks are Universal Approximators”. In: *Neural Networks* 2 (2010), pp. 359–366. URL: `http://deeplearning.cs.cmu.edu/pdfs/Kornick{\_}et{\_}al.pdf`.

[HOT06] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. “A Fast Learning Algorithm for Deep Belief Nets”. In: *Neural Computation* 18.7 (2006), pp. 1527–1554. ISSN: 0899-7667. DOI: `10.1162/neco.2006.18.7.1527`. URL: `http://www.mitpressjournals.org/doi/10.1162/neco.2006.18.7.1527`.

[HS13] Michiel Hermans and Benjamin Schrauwen. “Training and Analyzing Deep Recurrent Neural Networks”. In: *NIPS* (2013), pp. 190–198. ISSN: 10495258. URL: `http://papers.nips.cc/paper/5166-training-and-analysing-deep-recurrent-neural-networks.pdf`.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory.” In: *Neural computation* 9.8 (1997), pp. 1735–80. ISSN: 0899-7667. DOI: `10.1162/neco.1997.9.8.1735`. arXiv: `1206.2944`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/9377276`.

[Ish+04] Kazuo Ishii et al. “Identification of motion with Echo State Network”. In: *Ocean '04 - MTS/IEEE Techno-Ocean '04: Bridges across the Oceans - Conference Proceedings* 3 (2004), pp. 1205–1210. DOI: `10.1109/oceans.2004.1405751`.

[IY17] Masanobu Inubushi and Kazuyuki Yoshimura. “Reservoir Computing Beyond Memory-Nonlinearity Trade-off”. In: *Scientific Reports* 7.1 (2017), p. 10199. ISSN: 2045-2322. DOI: `10.1038/s41598-017-10257-6`. URL: `http://www.nature.com/articles/s41598-017-10257-6`.

[Jae+07] Herbert Jaeger et al. “Optimization and applications of echo state networks with leaky- integrator neurons”. In: *Neural Networks* 20.3 (2007), pp. 335–352. ISSN: 08936080. DOI: `10.1016/j.neunet.2007.04.016`.

[Jae01] Herbert Jaeger. “The echo state approach to analysing and training recurrent neural networks”. In: *GMD Report 148* (2001). URL: `http://minds.jacobs-university.de/sites/default/files/uploads/papers/EchoStatesTechRep.pdf`.

[Jae02a] Herbert Jaeger. “Adaptive Nonlinear System Identification with Echo State Networks”. In: *Advances in Neural Information Processing Systems (NIPS)* (2002), pp. 593–600. ISSN: 10495258. URL: `http://books.nips.cc/nips15.html`.

[Jae02b] Herbert Jaeger. *Short term memory in echo state networks.* Tech. rep. 2002.

[JH04] Herbert Jaeger and Harald Haas. “Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication”. In: *Science* 304.5667 (2004), pp. 78–80. ISSN: 0036-8075. DOI: `10.`

1126/science.1091277. arXiv: arXiv:1011.1669v3. URL: http://www.sciencemag.org/cgi/doi/10.1126/science.1091277.

[JL84] William B. Johnson and Joram Lindenstrauss. "Extensions of Lipschitz mappings into a Hilbert space". In: *Contemporary Mathematics* 26.January 1984 (1984), pp. 189–206. DOI: 10.1090/conm/026/737400. URL: http://www.ams.org/conm/026/.

[Kaw+17] Yuji Kawai et al. "Echo in a small-world reservoir: Time-series prediction using an economical recurrent neural network". In: *2017 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*. IEEE, 2017, pp. 126–131. ISBN: 978-1-5386-3715-9. DOI: 10.1109/DEVLRN.2017.8329797. URL: http://ieeexplore.ieee.org/document/8329797/.

[KCP86] M.M. Konstantinov, N.D. Christov, and P.Hr. Petkov. "On the Stability of Linear Stochastic Systems with Additive Noise". In: *IFAC Proceedings Volumes* 19.5 (1986), pp. 157–160. ISSN: 14746670. DOI: 10.1016/S1474-6670(17)59785-2. URL: http://linkinghub.elsevier.com/retrieve/pii/S1474667017597852.

[KJFF15] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. "Visualizing and Understanding Recurrent Networks". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8689 LNCS.PART 1 (2015), pp. 818–833. ISSN: 16113349. arXiv: 1506.02078. URL: http://arxiv.org/pdf/1506.02078.pdfhttp://arxiv.org/abs/1506.02078.

[KPA19] Yuji Kawai, Jihoon Park, and Minoru Asada. "A small-world topology enhances the echo state property and signal propagation in reservoir computing". In: *Neural Networks* 112 (2019), pp. 15–23. ISSN: 18792782. DOI: 10.1016/j.neunet.2019.01.002. URL: https://doi.org/10.1016/j.neunet.2019.01.002.

[KR16] Apoorva Khare and Bala Rajaratnam. *The Khinchin-Kahane inequality and Banach space embeddings for metric groups*. 2016. arXiv: 1610.03037. URL: http://arxiv.org/abs/1610.03037.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[LB16] Thomas Laurent and James von Brecht. *A recurrent neural network without chaos*. 2016. arXiv: 1612.06212. URL: http://arxiv.org/abs/1612.06212.

[Lea94] J. J. Leader. "The norms of powers of matrices with unit spectral radius". In: *Applied Mathematics Letters* 7.2 (1994), pp. 15–17. ISSN: 08939659. DOI: 10.1016/0893-9659(94)90023-X.

[Lip+15] Zachary C. Lipton et al. *Learning to Diagnose with LSTM Recurrent Neural Networks*. 2015. DOI: 10.14722/ndss.2015.23268. arXiv: 1511.03677. URL: http://arxiv.org/abs/1511.03677.

[Liu+12] Xiang Liu et al. "Performance evaluation of new echo state networks based on complex network". In: *Journal of China Universities of Posts and Telecommunications* 19.1 (2012), pp. 87–93. ISSN: 10058885. DOI: 10.1016/S1005-8885(11)60232-X. URL: http://dx.doi.org/10.1016/S1005-8885(11)60232-X.

[Liu+18] Xuanlin Liu et al. "Analysis of Memory Capacity for Deep Echo State Networks". In: *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (2018), pp. 443–448. DOI: 10.1109/ICMLA.2018.00072.

[LJ09] Mantas Lukoševičius and Herbert Jaeger. "Reservoir computing approaches to recurrent neural network training". In: *Computer Science Review* 3.3 (2009), pp. 127–149. ISSN: 15740137. DOI: 10.1016/j.cosrev.2009.03.005. URL: http://minds.jacobs-university.de/sites/default/files/uploads/papers/2261{_}LukoseviciusJaeger09.pdf.

[Luk12] M Lukoševičius. "A practical guide to applying echo state networks". In: *Neural Networks: Tricks of the Trade, Reloaded* (2012), pp. 659–686. ISSN: 03029743. DOI: 10.1007/978-3-642-35289-8-36. URL: http://minds.jacobs-university.de/sites/default/files/uploads/papers/PracticalESN.pdf.

[LYS09] Xiaowei Lin, Zehong Yang, and Yixu Song. "Short-term stock price prediction based on echo state networks". In: *Expert Systems with Applications* 36.3 PART 2 (2009), pp. 7313–7317. ISSN: 09574174. DOI: 10.1016/j.eswa.2008.09.049. URL: http://dx.doi.org/10.1016/j.eswa.2008.09.049.

[Maa+02] Wolfgang Maass et al. “Real-time computing without stable states: a new framework for neural computation based on perturbations.” In: *Neural computation* 14.11 (2002), pp. 2531–60. ISSN: 0899-7667. DOI: 10.1162/089976602760407955. URL: http://www.ncbi.nlm.nih.gov/pubmed/12433288.

[MHW17] Zeeshan Khawar Malik, Amir Hussain, and Qingming Jonathan Wu. “Multilayered Echo State Machine: A Novel Architecture and Algorithm”. In: *IEEE Transactions on Cybernetics* 47.4 (2017), pp. 946–959. ISSN: 21682267. DOI: 10.1109/TCYB.2016.2533545.

[MJ13] G. Manjunath and H. Jaeger. “Echo State Property Linked to an Input: Exploring a Fundamental Characteristic of Recurrent Neural Networks”. In: *Neural Computation* 25.3 (2013), pp. 671–696. ISSN: 0899-7667. DOI: 10.1162/NECO_a_00411. URL: http://www.mitpressjournals.org/doi/10.1162/NECO{_}a{_}00411.

[Mon+14] Guido Montúfar et al. “On the number of linear regions of deep neural networks”. In: *Advances in Neural Information Processing Systems* 4.January (2014), pp. 2924–2932. ISSN: 10495258. arXiv: 1402.1869.

[MS11] James Martens and Ilya Sutskever. “Learning recurrent neural networks with Hessian-free optimization”. In: *Proceedings of the 28th International Conference on Machine Learning, ICML 2011* (2011), pp. 1033–1040. URL: http://www.icml-2011.org/papers/532{_}icmlpaper.pdf.

[MSC17] Qianli Ma, Lifeng Shen, and Garrison W Cottrell. *Deep-ESN: A Multiple Projection-encoding Hierarchical Reservoir Computing Framework.* 2017. arXiv: 1711.05255. URL: http://arxiv.org/abs/1711.05255.

[Ng+15] Joe Yue Hei Ng et al. “Beyond short snippets: Deep networks for video classification”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 07-12-June (2015), pp. 4694–4702. ISSN: 10636919. DOI: 10.1109/CVPR.2015.7299101. arXiv: 1503.08909.

[O’T13] Garson O’Toole. *Quote Investigator: It’s Difficult to Make Predictions, Especially About the Future.* 2013. URL: https://quoteinvestigator.com/2013/10/20/no-predict/ (visited on 11/14/2019).

[Oli06] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[OXP07] Mustafa C Ozturk, Dongming Xu, and Jose C. Principe. "Analysis and Design of Echo State Networks for Function Approximation". In: *Neural Computation* 19.1 (2007), pp. 111–138.

[PA18] Kameleh Nassiri Pirbazari and Mehdi Azari. "The order of minimal realization of Jordan canonical form systems". In: *Boletim da Sociedade Paranaense de Matematica* 36.3 (2018), pp. 81–88. ISSN: 21751188. DOI: 10.5269/bspm.v36i3.23030.

[Pas+13] Razvan Pascanu et al. "How to Construct Deep Recurrent Neural Networks". In: *arXiv preprint arXiv:1312.6026* (2013), pp. 1–10. arXiv: 1312.6026. URL: http://arxiv.org/abs/1312.6026.

[PMB12] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. "On the difficulty of training recurrent neural networks". In: *Proceedings of The 30th International Conference on Machine Learning* 2 (2012), pp. 1310–1318. DOI: 10.1109/72.279181. URL: http://jmlr.org/proceedings/papers/v28/pascanu13.pdf.

[Qia+17] Junfei Qiao et al. "Growing Echo-State Network With Multiple Subreservoirs". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.2 (2017), pp. 391–404. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2016.2514275. URL: http://ieeexplore.ieee.org/document/7386673/.

[Riv12] Omar Rivasplata. *Subgaussian random variables : An expository note*. 2012. URL: http://www.mendeley.com/c/6521046371/p/7256383/rivasplata-2012-subgaussian-random-variables--an-expository-note/{\%}0Apapers3://publication/uuid/F630DD28-8135-4949-93F1-4077219E26DB.

[Rod12] Ali Rodan. "Architectural designs of Echo State Network". In: May (2012), p. 147. URL: http://etheses.bham.ac.uk/3610/{\%}0Ahttp://etheses.bham.ac.uk/3610/1/Alrodan12PhD.pdf.

[Ros58] F Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6 (1958), pp. 386–408. ISSN: 0033-295X. DOI: 10.1037/h0042519.

[RR08] Ali Rahimi and Benjamin Recht. “Uniform Approximation of Functions with Random Bases”. In: *46th Annual Allerton Conference on Communication, Control, and Computing* (2008).

[RR09] Ali Rahimi and B Recht. “Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning”. In: *Advances in Neural Information Processing Systems* (2009). URL: `http://papers.nips.cc/paper/3495-weighted-sums-of-random-kitchen-sinks-replacing-minimization-with-randomization-in-learning`.

[RT11a] Ali Rodan and Peter Tino. “Minimum Complexity Echo State Network”. In: *IEEE Transactions on Neural Networks* 22.1 (2011), pp. 131–144. ISSN: 1045-9227. DOI: `10.1109/TNN.2010.2089641`. URL: `http://ieeexplore.ieee.org/document/5629375/`.

[RT11b] Ali Rodan and Peter Tino. “Negatively Correlated Echo State Networks”. In: 1.April (2011), pp. 27–29.

[Rug96] Wilson J Rugh. *Linear Systems Theory*. 2nd. Prentice-Hall, 1996. ISBN: 9780262232586.

[RV08] Mark Rudelson and Roman Vershynin. “The Littlewood-Offord problem and invertibility of random matrices”. In: *Advances in Mathematics* 218.2 (2008), pp. 600–633. ISSN: 00018708. DOI: `10.1016/j.aim.2008.01.010`. arXiv: `0703503v2 [arXiv:math]`.

[Sch15] Jürgen Schmidhuber. “Deep Learning in neural networks: An overview”. In: *Neural Networks* 61 (2015), pp. 85–117. ISSN: 18792782. DOI: `10.1016/j.neunet.2014.09.003`. arXiv: `1404.7828`.

[Sch92] Jürgen Schmidhuber. “Learning Complex, Extended Sequences Using the Principle of History Compression”. In: *Neural Computation* 4.2 (1992), pp. 234–242. ISSN: 0899-7667. DOI: `10.1162/neco.1992.4.2.234`. URL: `http://www.mitpressjournals.org/doi/10.1162/neco.1992.4.2.234`.

[SH07] Mark D. Skowronski and John G. Harris. “Noise-robust automatic speech recognition using a predictive echo state network”. In: *IEEE Transactions on Audio, Speech and Language Processing* 15.5 (2007), pp. 1724–1730. ISSN: 15587916. DOI: `10.1109/TASL.2007.896669`.

[Sie95] H. T. Siegelmann. “Computation Beyond the Turing Limit”. In: *Science* 268.5210 (1995), pp. 545–548. ISSN: 0036-8075. DOI: 10.1126/science.268.5210.545. URL: http://www.sciencemag.org/cgi/doi/10.1126/science.268.5210.545.

[Sil00] John R. Silvester. “Determinants of Block Matrices”. In: *The Mathematical Gazette* 84.501 (2000), p. 460. ISSN: 00255572. DOI: 10.2307/3620776.

[SL09] Friedhelm Schwenker and Amr Labib. “Echo state networks and neural network ensembles to predict sunspots activity”. In: *ESANN 2009, 17th European Symposium on Artificial Neural Networks.* 2009.

[SS92] Hava T. Siegelmann and Eduardo D. Sontag. “On the computational power of neural nets”. In: *Proceedings of the fifth annual workshop on Computational learning theory - COLT ’92.* Vol. 50. 1. New York, New York, USA: ACM Press, 1992, pp. 440–449. ISBN: 089791497X. DOI: 10.1145/130385.130432. URL: https://linkinghub.elsevier.com/retrieve/pii/S0022000085710136http://portal.acm.org/citation.cfm?doid=130385.130432.

[STM18] Mohamad Kazem Shirani Faradonbeh, Ambuj Tewari, and George Michailidis. “Finite time identification in unstable linear systems”. In: *Automatica* 96.i (2018), pp. 342–353. ISSN: 00051098. DOI: 10.1016/j.automatica.2018.07.008. arXiv: 1710.01852.

[Sun+12] Xiao-chuan Sun et al. “Modeling deterministic echo state network with loop reservoir”. In: *Journal of Zhejiang University SCIENCE C* 13.9 (2012), pp. 689–701. ISSN: 1869-1951. DOI: 10.1631/jzus.C1200069. URL: http://link.springer.com/10.1631/jzus.C1200069.

[TCB04] P. Tino, M. Cernansky, and L. Benuskova. “Markovian Architectural Bias of Recurrent Neural Networks”. In: *IEEE Transactions on Neural Networks* 15.1 (2004), pp. 6–15. ISSN: 1045-9227. DOI: 10.1109/TNN.2003.820839. URL: https://www.cs.bham.ac.uk/{~}pxt/PAPERS/rnn.arch.bias.pdf.

[Tee05] Garry J Tee. “Eigenvectors of block circulant and alternating circulant matrices”. In: *Res. Lett. Inf. Math. Sci.* 8 (2005), pp. 123–142. ISSN: 1175-2777. URL: http://iims.massey.ac.nz/research/letters/volume8/tee/tee.pdf.

[Tel16] Matus Telgarsky. "Benefits of depth in neural networks". In: *Journal of Machine Learning Research* 49.June (2016), pp. 1517–1539. ISSN: 15337928. arXiv: 1602.04485.

[Tin17] Peter Tino. "Fisher Memory of Linear Wigner Echo State Networks". In: *European Symposium on Artificial Neural Networks (ESANN)* April (2017), pp. 26–28. URL: https://pdfs.semanticscholar.org/fb1a/48244d52f709f762e92a54c8be992fcc4547.pdf.

[Tom82] B. Tomaszewski. "Two remarks on the Khintchine-Kahane inequality". In: *Colloquium Mathematicum* 46.2 (1982), pp. 283–288. ISSN: 0010-1354. DOI: 10.4064/cm-46-2-283-288. URL: http://www.impan.pl/get/doi/10.4064/cm-46-2-283-288.

[TR13] Peter Tino and Ali Rodan. "Short term memory in input-driven linear dynamical systems". In: *Neurocomputing* 112 (2013), pp. 58–63. ISSN: 09252312. DOI: 10.1016/j.neucom.2012.12.041. URL: https://linkinghub.elsevier.com/retrieve/pii/S0925231213001768.

[Tro16] Joel A. Tropp. "The Expected Norm of a Sum of Independent Random Matrices: An Elementary Approach". In: June. 2016, pp. 173–202. DOI: 10.1007/978-3-319-40519-3_8. URL: http://arxiv.org/abs/1506.04711http://link.springer.com/10.1007/978-3-319-40519-3{_}8.

[TT90] Ruey S. Tsay and George C. Tiao. "Asymptotic Properties of Multivariate Nonstationary Processes with Applications to Autoregressions". In: *The Annals of Statistics* 18.1 (1990), pp. 220–250. ISSN: 0090-5364. DOI: 10.1214/aos/1176347499. URL: http://projecteuclid.org/euclid.aos/1176347499.

[TTC14] Fengzhen Tang, Peter Tino, and Huanhuan Chen. "Learning the deterministically constructed Echo State Networks". In: *Proceedings of the International Joint Conference on Neural Networks* (2014), pp. 77–83. DOI: 10.1109/IJCNN.2014.6889714.

[Use10] User17762. *Prove that the eigenvalues of a block matrix are the combined eigenvalues of its blocks*. Mathematics Stack Exchange. 2010. URL: https://math.stackexchange.com/q/21456.

[Wai15] Martin Wainwright. "Chapter 2 – Basic tail and concentration bounds". In: *210B Lecture Notes* University (2015). URL: https://www.stat.berkeley.edu/{~}mjwain/stat210b/Chap2{_}TailBounds{_}Jan22{_}2015.pdf.

[Wer90] Paul J. Werbos. "Backpropagation through time: What it does and how to do it". In: *Proceedings of the IEEE* 78.9039172 (1990), pp. 1550–1560.

[WJY15] Heshan Wang, Jian Huang, and Xuefeng Yan. "Improved cycle reservoir with regular jump networks with simple disjunction algorithm". In: *2015 Chinese Automation Congress (CAC)*. 1. IEEE, 2015, pp. 809–814. ISBN: 978-1-4673-7189-6. DOI: 10.1109/CAC.2015.7382609. URL: http://ieeexplore.ieee.org/document/7382609/.

[WLS04] Olivia L. White, Daniel D. Lee, and Haim Sompolinsky. "Short-Term Memory in Orthogonal Neural Networks". In: *Physical Review Letters* 92.14 (2004), p. 148102. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.92.148102. URL: https://link.aps.org/doi/10.1103/PhysRevLett.92.148102.

[XYH07] Yanbo Xue, Le Yang, and Simon Haykin. "Decoupled echo state networks with lateral inhibition". In: *Neural Networks* 20.3 (2007), pp. 365–376. ISSN: 08936080. DOI: 10.1016/j.neunet.2007.04.014.

[YBK88] Y. Q. Yin, Z. D. Bai, and P. R. Krishnaiah. "On the limit of the largest eigenvalue of the large dimensional sample covariance matrix". In: *Probability Theory and Related Fields* 78.4 (1988), pp. 509–521. ISSN: 01788051. DOI: 10.1007/BF00353874.

[YJK12] Izzet B. Yildiz, Herbert Jaeger, and Stefan J. Kiebel. "Re-visiting the echo state property". In: *Neural Networks* 35 (2012), pp. 1–9. ISSN: 08936080. DOI: 10.1016/j.neunet.2012.07.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S0893608012001852.

[YN15] Sumeth Yuenyong and Akinori Nishihara. "Evolutionary pre-training for CRJ-type reservoir of echo state networks". In: *Neurocomputing* 149.PC (2015), pp. 1324–1329. ISSN: 18728286. DOI: 10.1016/j.neucom.2014.08.065. URL: http://dx.doi.org/10.1016/j.neucom.2014.08.065.

[Zha+16] Saizheng Zhang et al. *Architectural Complexity Measures of Recurrent Neural Networks.* 2016. arXiv: `1602.08210`. URL: `http://arxiv.org/abs/1602.08210`.

[Zha03] G.Peter Zhang. "Time series forecasting using a hybrid ARIMA and neural network model". In: *Neurocomputing* 50 (2003), pp. 159–175. ISSN: 09252312. DOI: `10.1016/S0925-2312(01)00702-0`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0925231201007020`.

[ZMW12] Bai Zhang, David J. Miller, and Yue Wang. "Nonlinear System Modeling With Random Matrices: Echo State Networks Revisited". In: *IEEE Transactions on Neural Networks and Learning Systems* 23.1 (2012), pp. 175–182. ISSN: 2162-237X. DOI: `10.1109/TNNLS.2011.2178562`. URL: `http://ieeexplore.ieee.org/document/6105577/`.

[ZW08] Bai Zhang and Yue Wang. "Echo state networks with decoupled reservoir states". In: *Proceedings of the 2008 IEEE Workshop on Machine Learning for Signal Processing, MLSP 2008* (2008), pp. 444–449. DOI: `10.1109/MLSP.2008.4685521`.

[Rum85] Ronald J Rumelhart, David E and Hinton, Geoffrey E and Williams. *Learning Internal Representations by Error Propagation.* Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science, 1985. DOI: `https://doi.org/10.1016/B978-1-4832-1446-7.50035-2`. URL: `http://www.csri.utoronto.ca/{~}hinton/absps/pdp8.pdf`.

Appendix A

Linear Algebra Background

In this Appendix, we review the pre-requisite linear algebra for the research chapters of the thesis.

A.1 Matrix Decompositions

Definition A.1.1 (Diagonalisable, eigendecomposition). *For $A \in \mathbb{R}^{M \times M}$, we say that A is diagonalisable if there exist non-singular matrix P and diagonal matrix Λ such that $A = P\Lambda P^{-1}$, where the columns of P are eigenvectors of A and Λ is the diagonal matrix whose ith diagonal entry is the eigenvalue to which the ith column of P corresponds. We refer to $P\Lambda P^{-1}$ as the Eigendecomposition of A.*

Not all matrices are diagonalisable, and therefore not every matrix has an eigendecomposition. However, there does exist a generalisation of this idea, the *Jordan Normal Form.*

Definition A.1.2 (Jordan Block, Jordan Matrix). *For $\lambda \in \mathbb{C}$ and $k \in \mathbb{N}$, we define the Jordan block $J_k(\lambda)$ as the $k \times k$ matrix of the form*

$$\begin{pmatrix} \lambda & 1 & 0 & \cdots & 0 \\ 0 & \lambda & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & 1 \\ 0 & 0 & 0 & \cdots & \lambda \end{pmatrix}.$$

We say that an $M \times M$ matrix is a Jordan matrix if it is of the form

$$\begin{pmatrix} J_{n_1}(\lambda_1) & 0 & \cdots & 0 \\ 0 & J_{n_2}(\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_{n_l}(\lambda_l) \end{pmatrix}$$

where each $J_{n_i}(\lambda_i)$ is a Jordan block.

Definition A.1.3 (Jordan Normal Form, Jordan Decomposition)**.** *For $A \in \mathbb{R}^{M\times M}$, we say that J is the Jordan normal form of A if J is a Jordan matrix and there exists non-singular $M \times M$ matrix P such that $A = PJP^{-1}$. We call JPJ^{-1} the Jordan decomposition of A.*

The number of Jordan blocks for a given eigenvalue is the *geometric multiplicity* of that eigenvalue. The *algebraic multiplicity* of that eigenvalue is the sum of the sizes of those blocks. In the eigendecomposition, columns of the matrix P are the eigenvectors of A; in the Jordan decomposition, the columns of P are *generalised eigenvectors* of A.

Definition A.1.4 (Rank of Generalised Eigenvectors)**.** *A vector $\mathbf{p} \in \mathbb{R}^M$ is said to be a generalised eigenvector of the matrix $A \in \mathbb{R}^M$ of rank r if*

$$(A - \lambda I)^r \mathbf{v} = 0$$

and

$$(A - \lambda I)^{r-1} \mathbf{v} \neq 0.$$

Consider an $M \times M$ Jordan matrix, with a Jordan block of size d running between the rows indexed $a + 1$ and b (inclusive) for some $0 \leq a < b \leq M$ with $a - b = d$. For $1 \leq i \leq d$, the $(a + i)$th column of P contains an eigenvector of rank i.

Proposition A.1.1. *Let $J_k(\lambda)$ be a Jordan block and $t \in \mathbb{N}$,*

$$(J_k(\lambda))^t = \begin{pmatrix} \lambda^t & \binom{t}{1}\lambda^{t-1} & \binom{t}{2}\lambda^{t-2} & \cdots & \binom{t}{k-1}\lambda^{t-k+1} \\ 0 & \lambda^t & \binom{t}{1}\lambda^{t-1} & \cdots & \binom{t}{k-2}\lambda^{t-k+2} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda^t & \binom{t}{1}\lambda^{t-1} \\ 0 & 0 & \cdots & 0 & \lambda^t \end{pmatrix} \tag{A.1}$$

Proof. See, for instance [GV96]. □

It is useful to note that for any fixed j, when $t \geq j$,

$$\begin{aligned} \binom{t}{j} &= \frac{t!}{j! \cdot (t-j)!} \\ &= \frac{t \cdot (t-1) \cdot \cdots \cdot (t-j+1)}{j!} \\ &= \frac{t^j + o(t^j)}{j!}, \end{aligned} \tag{A.2}$$

where the last line is obtained by noting that there are j terms in the product in the line above, and reasoning about what the expanded product would like. Therefore the upper triangular entries of $(J_k(\lambda))^t$ are of the form $p_{ij}(t) \cdot \lambda^t$, where p_{ij} is a polynomial of degree at most $k-1$.

The following proposition is used in Chapter 4. The proof we provide is adapted from [Use10].

Proposition A.1.2. *Define a block triangular matrix*

$$M = \begin{pmatrix} A_{11} & \mathbf{0} & \cdots & \mathbf{0} \\ A_{21} & A_{22} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \cdots & A_{nn} \end{pmatrix},$$

$\lambda \in \mathbb{C}$ is an eigenvalue of M if and only if it is also an eigenvalue of of at least one of $A_{11}, A_{22}, \ldots, A_{nn}$.

Proof. By Lemma 2 of [Sil00], we have that

$$\det\left(\begin{pmatrix} A & \mathbf{0} \\ B & C \end{pmatrix}\right) = \det(AC) = \det(A) \cdot \det(C)$$

We omit the details, but this can naturally be extended to more general block triangular matrices using induction in the obvious way, giving $\det(M) = \prod_{i=1}^{n} \det(A_{ii})$.

We can now consider, for some λ the block-triangular matrix $(M - \lambda I)$. λ is an eigenvalue of M if and only if $\det(M - \lambda I) = 0$. Equivalently the condition can be expressed as $\prod_{i=1}^{L} \det(A_{ii} - \lambda I) = 0$. Clearly this will happen if and only if one of the factors in the product is zero, which in turn only occurs if λ is an eigenvalue for some A_{ii}. □

A.2 Useful Inequalities

Chapter 6 is frequently be concerned with the magnitude of certain vector quantities, particularly in relation to their norms. As such, in the chapter, we make use of several standard inequalities. For completeness, we state the necessary inequalities here.

Theorem A.2.1 (Triangle Inequality). *Let $|||\cdot|||$ be a norm on the space $\mathbb{R}^M$. For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$, we have*

$$|||\mathbf{x} + \mathbf{y}||| \leq |||\mathbf{x}||| + |||\mathbf{y}|||.$$

This theorem is tautologically true, since it describes a necessary condition for $|||\cdot|||$ being a norm. Through manipulations of the triangle inequality, we are able to derive the reverse triangle inequality, which we state now.

Theorem A.2.2 (Reverse Triangle Inequality). *Let $|||\cdot|||$ be a norm on the space $\mathbb{R}^M$. For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$, we have*

$$\left| |||\mathbf{x}||| - |||\mathbf{y}||| \right| \leq |||\mathbf{x} - \mathbf{y}|||.$$

Theorem A.2.3 (Jensen's Inequality). *If X be a random variable taking values in $\mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ is a convex function, then*

$$\mathbb{E}\left[g(X)\right] \geq g\left(\mathbb{E}\left[X\right]\right).$$

Proof. See, for example, [HJ85] □

Theorem A.2.4 (Equivalence of Norms). *Let $\|\cdot\|_a$ and $\|\cdot\|_b$ be two norms on a finite dimensional vector space V. There exist constants $0 < C_1 \leq C_2$ such that for all $\mathbf{x} \in V$,*

$$C_1 \cdot \|\mathbf{x}\|_a \leq \|\mathbf{x}\|_b \leq C_2 \cdot \|\mathbf{x}\|_a .$$

Proof. See, for example, [HJ85] □

Next, we consider a couple of results bounding the expectation of norms of random variables.

Theorem A.2.5. *Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ be n independent random vectors with mean zero and finite expected norm, and let $r_1, \ldots, r_n$ be n independent Rademacher random variables,*

$$\mathbb{E}\left[\left\|\sum_{i=1}^{n} \boldsymbol{x}_i\right\|\right] \geq \frac{1}{2} \cdot \mathbb{E}\left[\left\|\sum_{i=1}^{n} r_i \boldsymbol{x}_i\right\|\right].$$

Proof. See [Tro16] □

Theorem A.2.6 (The Khinchin-Kahane Inequality). *For all $p, q \in [1, \infty)$, there exists a constant $C_{p,q} > 0$ depending only upon p, q such that for all $n \in \mathbb{N}$, for all finite sets of vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^M$ and independent Rademacher variables $r_1, r_2, \ldots, r_n$,*

$$\mathbb{E}\left[\left\|\sum_{i=1}^{n} r_i \mathbf{x}_i\right\|^q\right]^{\frac{1}{q}} \leq C_{p,q} \cdot \mathbb{E}\left[\left\|\sum_{i=1}^{n} r_i \mathbf{x}_i\right\|^p\right]^{\frac{1}{p}}.$$

Proof. See, for instance [Tom82] (though the above formulation of the theorem is taken from [KR16]). □

Re-arranging, we can also get the inequality

$$\mathbb{E}\left[\left\|\sum_{i=1}^{n} r_i \mathbf{x}_i\right\|^p\right]^{\frac{1}{p}} \geq \frac{1}{C_{p,q}} \cdot \mathbb{E}\left[\left\|\sum_{i=1}^{n} r_i \mathbf{x}_i\right\|^q\right]^{\frac{1}{q}}.$$

Though the theorem is applied to fixed vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n$, it is trivial to extend it to the expectation of random vectors. Suppose that $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are random vectors in $\mathbb{R}^M$. We can use the fact that if $f(x) \geq g(x)$ for all values of x then

$\int f(x)\mathrm{d}x \geq \int g(x)\mathrm{d}x$, whenever the two integrals are over the same region of the space, to take the expectation not just over each r_i, but also over the $\boldsymbol{x}_i$, yielding

$$\mathbb{E}\left[\left\|\sum_{i=1}^{n} r_i \boldsymbol{x}_i\right\|^p\right]^{\frac{1}{p}} \geq \frac{1}{C_{p,q}} \cdot \mathbb{E}\left[\left\|\sum_{i=1}^{n} r_i \boldsymbol{x}_i\right\|^q\right]^{\frac{1}{q}}.$$

Lemma A.2.1. *Let $A \in \mathbb{R}^{M\times M}$ be a matrix with spectral radius $\rho(A)$. For any $\epsilon > 0$ there exists a sub-multiplicative matrix norm $|||\cdot|||$ such that $\rho(A) \leq |||A||| \leq \rho(A) + \epsilon$.*

Proof. See, for instance, Lemma 5.6.10 from [HJ85] □

A.3 Gelfand's Formula

In Chapter 5 and and Chapter 6, we make use of the following lemma:

Lemma A.3.1. *Let A be a matrix such that $\rho(A) < 1$. For all γ satisfying $\rho(A) < \gamma$, there exists $k_0 \in \mathbb{N}$ such that, for all $k > k_0$, $\left\|A^k\right\| < \gamma^k$.*

Proof. Let $A = PJP^{-1}$ where J is the Jordan form of A, we can write, by sub-multiplicativity of the norm and equivalence of norms, we can write

$$\begin{aligned}\left\|A^k\right\| &\leq \|P\| \cdot \left\|J^k\right\| \cdot \left\|P^{-1}\right\| \\ &= C_1' \cdot \left\|J^k\right\| \\ &\leq C_1 \cdot \left\|J^k\right\|_F,\end{aligned}$$

where $\|\cdot\|_F$ is the Frobenius norm[1], $C_1' \overset{\text{def}}{=} \|P\| \cdot \|P^{-1}\|$ and C_1 is some positive constant. Since all elements of J^k grow at a rate bounded by $C_2 \cdot k^{M-1} \cdot \rho(W)^k$ (this can be seen by using A.2 to determine the degree of the polynomials bounding the combination functions in Equation A.1), for some $C_2 > 0$, we have for any

[1]That is, the norm defined as $\|X\|_F = \sqrt{\sum_{i,j=1}^{M} |(X)_{ij}|^2}$ for $X \in \mathbb{X}^{M\times M}$

$\epsilon > 0$, for all sufficiently large k

$$\begin{aligned}\left\|A^k\right\| &\leq C_1 \cdot \left\|J^k\right\|_F \\ &= C_1 \cdot \left(\sum_{i=1}^{M}\sum_{j=1}^{M}(\left|J^k\right|)_{ij}^2\right)^{\frac{1}{2}} \\ &\leq C_1 \cdot \left(M^2 \cdot C_2 \cdot k^{M-1} \cdot \rho(A)^{2\cdot k}\right)^{\frac{1}{2}} \\ &\leq C_1 \cdot \left(M^2 \cdot C_2 \cdot (\rho(A)+\epsilon)^{2\cdot k}\right)^{\frac{1}{2}} \\ &= C_1 \cdot M \cdot \sqrt{C_2} \cdot (\rho(A)+\epsilon)^k\end{aligned}$$

For any constant $\gamma > \rho(A)$, we can choose $\epsilon > 0$ such that $\rho(A) + \epsilon < \gamma$ and therefore have that for such ϵ there exists k_0 such that for all $k > k_0$, $\gamma^k > C_1 \cdot M \cdot (\rho(A) + \epsilon)^k$, completing the proof. □

Appendix B

Supplementary Material for Chapter 4

In this appendix, we state and prove intermediate results required for Theorem 4.3.1.

Lemma B.0.1. *Let D be a $d \times M$ matrix of the form*

$$D \overset{\text{def}}{=} \begin{pmatrix} J_d(\lambda)\mathbf{v} & J_d(\lambda)^2\mathbf{v} & \cdots & J_d(\lambda)^M\mathbf{v} \end{pmatrix}$$

Where $\mathbf{v} \in \mathbb{R}^d$ and $J_d(\lambda)$ is the Jordan block of size d with eigenvalue λ (see Appendix A.1 for more detail). For $2 \leq k \leq M$ and $1 \leq i \leq d$,

$$(D)_{ik} = (D)_{(i+1)(k-1)} + \lambda(D_{i(k-1)}),$$

where we define $(D)_{(d+1)k} = 0$ for all k.

Proof. For all k,

$$(J_d(\lambda))^k\mathbf{v} = \begin{pmatrix} \lambda^k & \binom{k}{1}\lambda^{k-1} & \binom{k}{2}\lambda^{k-2} & \cdots & \binom{k}{d-1}\lambda^{k-d+1} \\ 0 & \lambda^k & \binom{k}{1}\lambda^{k-1} & \cdots & \binom{k}{d-2}\lambda^{k-d+2} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda^k & \binom{k}{1}\lambda^{k-1} \\ 0 & 0 & \cdots & 0 & \lambda^k \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{d-1} \\ v_d \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{j=0}^{d-1} \lambda^{k-j} \binom{k}{j} v_{1+j} \\ \sum_{j=0}^{d-2} \lambda^{k-j} \binom{k}{j} v_{2+j} \\ \vdots \\ \sum_{j=0}^{1} \lambda^{k-j} \binom{k}{j} v_{d-1+j} \\ \lambda^k v_d \end{pmatrix}$$

So, in general we have

$$(D)_{ik} = \sum_{j=0}^{d-i} \lambda^{k-j} \cdot \binom{k}{j} \cdot v_{i+j}.$$

Therefore

$$\begin{aligned} (D)_{ik} &= \sum_{j=0}^{d-i} \lambda^{k-j} \cdot \binom{k}{j} \cdot v'_{i+j} \\ &= \sum_{j=0}^{d-i} \lambda^{k-j} \cdot \binom{k-1}{j-1} \cdot v'_{i+j} + \sum_{j=0}^{d-i} \lambda^{k-j} \cdot \binom{k-1}{j} \cdot v'_{i+j} \\ &= \sum_{j=-1}^{d-i-1} \lambda^{k-(j+1)} \cdot \binom{k-1}{j} \cdot v'_{i+j+1} + \lambda \sum_{j=0}^{d-i} \lambda^{(k-1)-j} \cdot \binom{k-1}{j} \cdot v'_{i+j} \\ &= \sum_{j=0}^{d-(i+1)} \lambda^{(k-1)-j} \cdot \binom{k-1}{j} \cdot v'_{(i+1)+j} + \lambda \sum_{j=0}^{d-i} \lambda^{(k-1)-j} \cdot \binom{k-1}{j} \cdot v'_{i+j} \\ &= D_{(i+1)(k-1)} + \lambda \cdot D_{i(k-1)}. \end{aligned}$$

We get second line in the above manipulations using the identity $\binom{k}{j} = \binom{k-1}{j-1} + \binom{k-1}{j}$. $\square$

Lemma B.0.2. *Let J be a matrix in Jordan form with at least two blocks corresponding to some eigenvalue $\lambda \neq 0$. Choose two such blocks J_1 and J_2 of effective sizes d_1 and d_2 (with respect to vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively) and $d_1 \leq d_2$. For any M, the rows of the matrix*

$$M_1 \overset{\text{def}}{=} \begin{pmatrix} J_1\mathbf{v}_1 & J_1^2\mathbf{v}_1 & \cdots & J_1^M\mathbf{v}_1 \end{pmatrix}$$

are in the row space of the matrix

$$M_2 \overset{\text{def}}{=} \begin{pmatrix} J_2\mathbf{v}_2 & J_2^2\mathbf{v}_2 & \cdots & J_2^M\mathbf{v}_2, \end{pmatrix}$$

Furthermore, the rank of that row space is d_2.

Proof. We begin by noting that for any positive integer k,

$$(M_1)_{ik} = (J_1^k \mathbf{v}_1)_i = \sum_{j=0}^{d_1-i} \lambda^{k-j} \binom{k}{j} (\mathbf{v}_1)_{j+i}$$

and similarly

$$(M_2)_{ik} = (J_2^k \mathbf{v}_2)_i = \sum_{j=0}^{d_2-i} \lambda^{k-j} \binom{k}{j} (\mathbf{v}_2)_{j+i}.$$

Our strategy is to show that the rows of the M_2, and the M_1 are both in the row-space of the matrix

$$\Gamma = \begin{pmatrix} \binom{1}{0}\lambda^1 & \binom{2}{0}\lambda^2 & \cdots & \binom{M}{0}\lambda^M \\ \binom{1}{1}\lambda^1 & \binom{2}{1}\lambda^1 & \ddots & \binom{2}{1}\lambda^M \\ \vdots & \ddots & \ddots & \vdots \\ \binom{1}{d_2-1}\lambda^1 & \cdots & \cdots & \binom{M}{d_2-1}\lambda^M \end{pmatrix}.$$

Note that we can write $(\Gamma)_{ik} = \binom{k}{j}\lambda^k$, and therefore we can write $(M_1)_{ik}$ as

$$\begin{aligned} (M_1)_{ik} &= \sum_{j=0}^{d-i} \lambda^{k-j} \cdot \binom{k}{j} \cdot (\mathbf{v}_2)_{i+j} \\ &= \sum_{j=0}^{d-i} \lambda^{-j} \cdot \left[\lambda^k \cdot \binom{k}{j}\right] \cdot (\mathbf{v}_2)_{i+j} \\ &= \sum_{j=0}^{d-i} \lambda^{-j} \cdot [(\Gamma)_{jk}] \cdot (\mathbf{v}_2)_{i+j}. \end{aligned}$$

This gives that we can write the ith row of M_1 as

$$(M_1)_{i\cdot} = \sum_{j=0}^{d_1-i} \lambda^{-j} \cdot \mathbf{v}_{i+j} \;\; \Gamma_{j\cdot},$$

and therefore every row of M_1 is in the row-space of Γ.

The same argument can be applied to M_2 to get that each of its rows are in the row-space of Γ, giving that both rows of M_1 and M_2 are in the row-space of Γ.

Noting that $(\mathbf{v}_1)_{d_2} \neq 0$, we can obtain that M_2 is rank d_2 via Hautus' lemma by considering a system with the recurrent weight matrix as Jordan block of size d_2 with an input vector $\mathbf{v} \in \mathbb{R}^{d_2}$ satisfying $(\mathbf{v})_{d_2} \neq 0$. Since all rows of M_2 are in Γ's row space, this means that $\text{rank}(\Gamma) \geq d_2$. Given that Γ is of size $d_2 \times M$, it must therefore be the case that $\text{rank}(\Gamma) = d_2$ $\square$

Appendix C

Supplementary Material for Chapter 5

This appendix contains supplementary material for Chapter 5. In Section C.1, we provide proof of Lemma 5.4.1. In Section C.2, we shown additional experiments examining the role of non-linearity in DeepESNs.

C.1 Proof of Lemma 5.4.1

In this section, we re-state and provide proof for Lemma 5.4.1.

Lemma 5.4.1. *Let $(\bar{W}, \bar{\mathbf{v}})$ be an L layer neural network with M hidden units per layer. For $1 \leq l \leq L$, let Q be a matrix satisfying $Q\left(\mathbb{E}\left[\mathbf{z}_t^{(l)}(\mathbf{z}_t^{(l)})^\intercal\right]\right)Q^\intercal = I$, where $Q = \Lambda^{-\frac{1}{2}}P^\intercal$ for an orthonormal matrix P and a full-rank diagonal matrix Λ, and let Q satisfy $\mathbb{E}\left[(Q\mathbf{z}_t^{(l)})_i u_t\right] = \delta_{i1}$, where δ_{ij} is the Kronecker delta. Defining $\tilde{\mathbf{z}} \stackrel{\text{def}}{=} Q\mathbf{z}$, if the input sequence has zero-mean and unit variance, for $k \geq 1$ the following equalities hold:*

$$\begin{aligned} MC_k^{(l)} &= \sum_{k=1}^{\infty} \mathbb{E}\left[(y_k^{(l)})^2\right] \\ &= \sum_{i=2}^{M+1} \mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right]^2. \end{aligned}$$

Where $y_k^{(l)}$ is the optimal linear reconstruction of u_{t-k} from the state $\mathbf{z}_t^{(l)}$.

Proof. To begin the proof, we use that by using the closed-form solution for linear regression we can expand the optimal reconstruction weights $\mathbf{w}_k^{(l)}$ to get

$$\begin{aligned} y_k &= (\mathbf{w}_k^{(l)})^\intercal \mathbf{z}_t^{(l)} \\ &= \left(\left(\mathbb{E}\left[\mathbf{z}_t^{(l)} \left(\mathbf{z}_t^{(l)}\right)^\intercal \right] \right)^{-1} \mathbb{E}\left[\mathbf{z}_t^{(l)} u_{t-k} \right] \right)^\intercal \mathbf{z}_t^{(l)}. \end{aligned} \tag{C.1}$$

Define $\tilde{\mathbf{p}}_k = Q\mathbb{E}\left[\mathbf{z}_t^{(l)} u_{t-k}\right]$, $\mathbf{p}_k = \mathbb{E}\left[\mathbf{z}_t^{(l)} - u_{t-k}\right]$ and $R = \mathbb{E}\left[\mathbf{z}_t^{(l)}\left(\mathbf{z}_t^{(l)}\right)^\intercal\right]$. From the definition of $Q = \Lambda^{-\frac{1}{2}} P^\intercal$, we get that $Q^{-1} = Q^\intercal \Lambda$. Using this, along with Equation C.1, gives

$$\begin{aligned} y_k &= (R^{-1}\mathbf{p}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= ((Q^{-1}QRQ^{-1}Q)^{-1}Q^{-1}Q\mathbf{p}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= ((Q^{-1}\underbrace{QRQ^T}_{=I}\Lambda Q)^{-1}Q^{-1}Q\mathbf{p}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= ((Q^{-1}\Lambda Q)^{-1}Q^{-1}\tilde{\mathbf{p}}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= (Q^{-1}\Lambda^{-1}QQ^{-1}\tilde{\mathbf{p}}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= (Q^{-1}\Lambda^{-1}\tilde{\mathbf{p}}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= (Q^\intercal\Lambda\Lambda^{-1}\tilde{\mathbf{p}}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= (Q^\intercal\tilde{\mathbf{p}}_k)^\intercal \mathbf{z}_t^{(l)} \\ &= (\tilde{\mathbf{p}}_k)^\intercal Q\mathbf{z}_t^{(l)} \\ &= (\tilde{\mathbf{p}}_k)^\intercal \tilde{\mathbf{z}}_t^{(l)} \\ &= \sum_{i=1}^{M+1} \mathbb{E}\left[\left(\tilde{\mathbf{z}}_t^{(l)}\right)_i \cdot u_{t-k}\right] (\tilde{\mathbf{z}}_t^{(l)})_i & \text{(C.2)} \\ &= \sum_{i=2}^{M+1} \mathbb{E}\left[\left(\tilde{\mathbf{z}}_t^{(l)}\right)_i \cdot u_{t-k}\right] (\tilde{\mathbf{z}}_t^{(l)})_i, & \text{(C.3)} \end{aligned}$$

Where the last equivalence comes from the assumption in the statement of the lemma about the construction of Q. From this, we can derive the second equality in the statement of the lemma, since by the above

$$\begin{aligned} \mathbb{E}\left[y_k^2\right] &= \mathbb{E}\left[\left(\sum_{i=2}^{M+1} \mathbb{E}\left[\left(\left(\tilde{\mathbf{z}}_t^{(l)}\right)_i \cdot u_{t-k}\right)\right] \cdot \left(\tilde{\mathbf{z}}_t^{(l)}\right)_i\right)^2\right] \\ &= \mathbb{E}\left[\sum_{i=2}^{M+1}\left(\left(\tilde{\mathbf{z}}_t^{(l)}\right)_i \cdot u_{t-k}\right)^2\right]. \end{aligned} \tag{C.4}$$

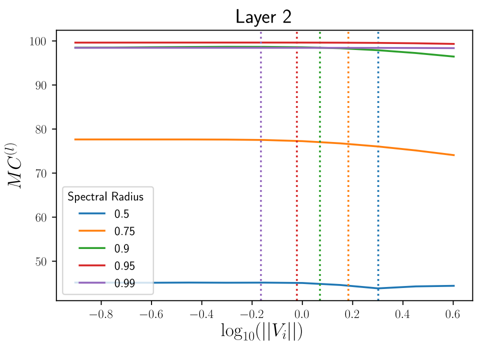
We get the second equality in the above manipulation by expanding the square and using that by definition of $\tilde{\mathbf{z}}_t$, $\mathbb{E}\left[(\tilde{\mathbf{z}}_t)_i \cdot (\tilde{\mathbf{z}}_t)_j\right] = 0$ for $i \neq j$.

For the first part of the lemma, plugging Equations C.3 and C.4 into the definition of $MC_k^{(l)}$, we get

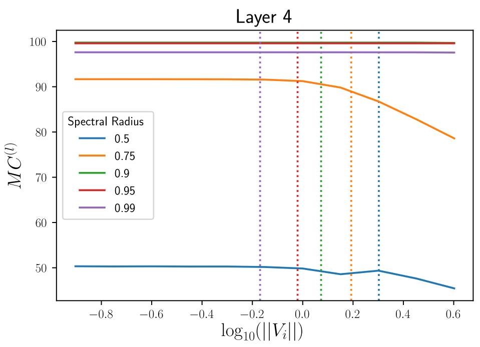
$$\begin{aligned}
MC_k^{(l)} &= \frac{\operatorname{cov}^2(y_k, u_{t-k})}{\operatorname{var}(y_k)\cdot\operatorname{var}(u_t)} \\
&= \frac{\left(\mathbb{E}\left[y_k \cdot u_{t-k}\right]\right)^2}{\mathbb{E}\left[y_k^2\right]\cdot\operatorname{var}(u_{t-k})} \\
&= \frac{\left(\sum_{i=2}^{M+1}\left(\mathbb{E}\left[\mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right](\tilde{\mathbf{z}}_t^{(l)})_i\right)\cdot u_{t-k}\right]\right)^2}{\mathbb{E}\left[\sum_{i=2}^{M+1}\left((\tilde{\mathbf{z}}_t{}^{(l)})_i \cdot u_{t-k}\right)^2\right]\cdot\operatorname{var}(u_{t-k})} \\
&= \frac{\left(\sum_{i=2}^{M+1}\left(\mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right]\mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right]\right)\right)^2}{\mathbb{E}\left[\sum_{i=2}^{M+1}\left((\tilde{\mathbf{z}}_t{}^{(l)})_i \cdot u_{t-k}\right)^2\right]\cdot\operatorname{var}(u_{t-k})} \\
&= \frac{\left(\sum_{i=2}^{M+1}\left(\mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right]\right)^2\right)^2}{\mathbb{E}\left[\sum_{i=2}^{M+1}\left((\tilde{\mathbf{z}}_t{}^{(l)})_i \cdot u_{t-k}\right)^2\right]\cdot\operatorname{var}(u_{t-k})} \\
&= \frac{\left(\sum_{i=2}^{M+1}\left(\mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right]\right)^2\right)}{\operatorname{var}(u_{t-k})} \\
&= \sum_{i=2}^{M+1}\left(\mathbb{E}\left[(\tilde{\mathbf{z}}_t^{(l)})_i \cdot u_{t-k}\right]\right)^2.
\end{aligned}$$

□

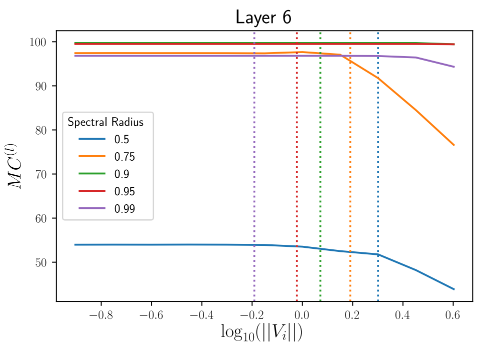
C.2 Additional Non-Linear Memory Capacity Experiments



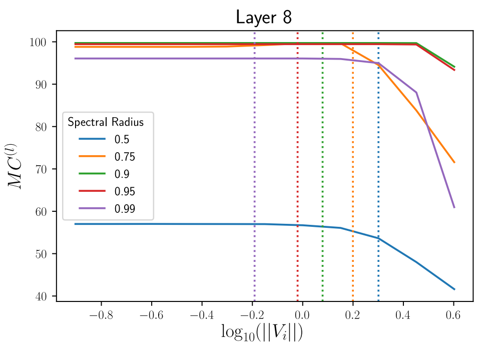
(a)



(b)



(c)



(d)

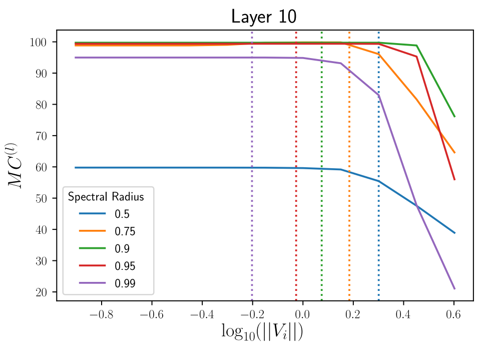
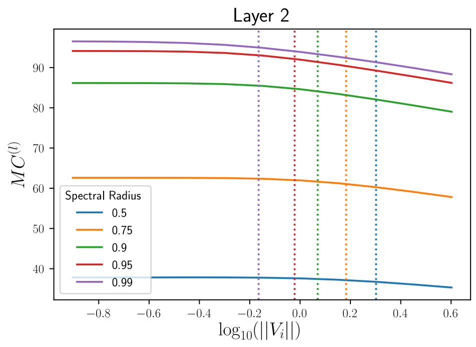
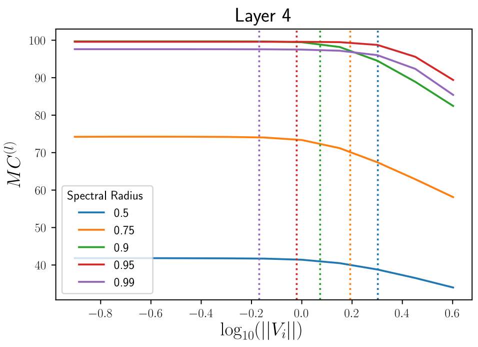


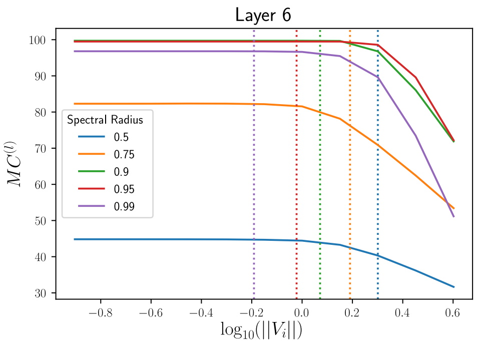
Figure C.1: Total memory capacity for network layers, varying recurrent spectral radius and feedforward norm size. Input weight vector scaled to satisfy $\|\mathbf{v}_i\| = 10^{-6}$



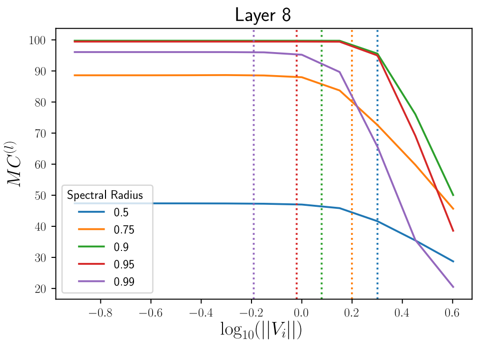
(a)



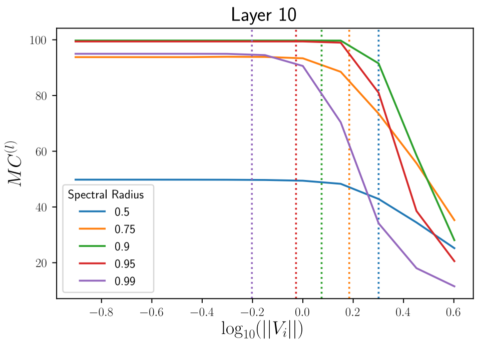
(b)



(c)

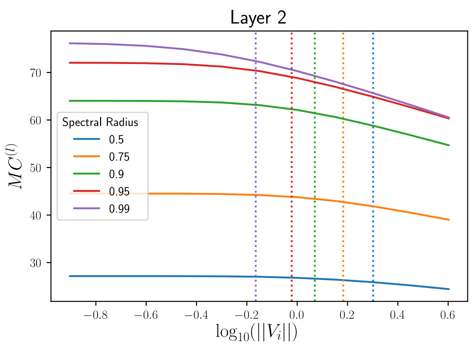


(d)

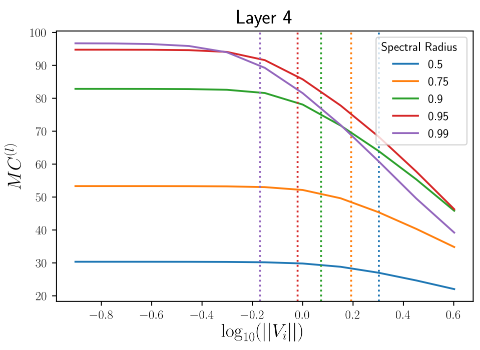


(e)

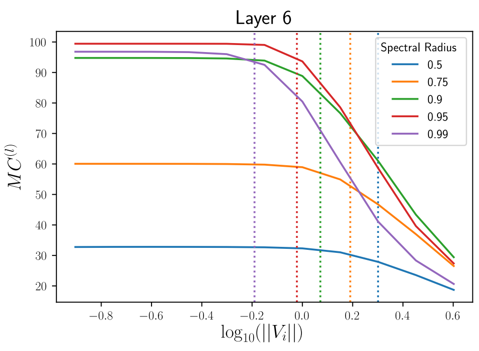
Figure C.2: Total memory capacity for network layers, varying recurrent spectral radius and feedforward norm size. Input weight vector scaled to satisfy $\|\mathbf{v}_i\| = 10^{-4}$



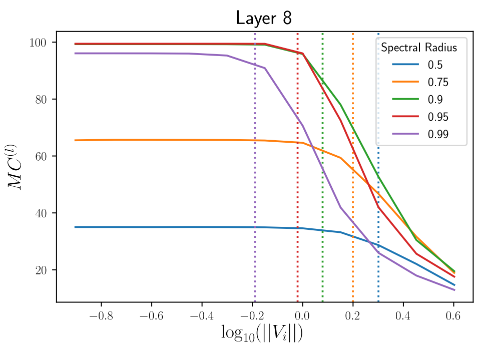
(a)



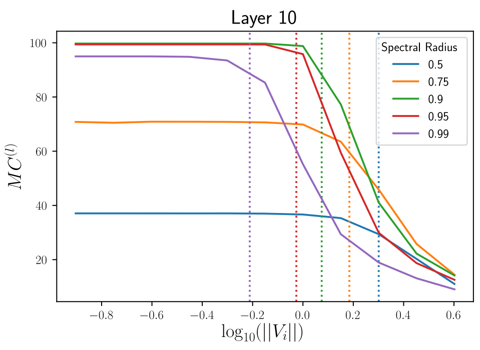
(b)



(c)



(d)



(e)

Figure C.3: Total memory capacity for network layers, varying recurrent spectral radius and feedforward norm size. Input weight vector scaled to satisfy $\|\mathbf{v}_i\| = 0.01$

C.3 Long Term Behaviour of Perturbations

In this appendix, we state and prove a result regarding the asymptotic behaviour of the vector $W^k\mathbf{v}$ as $k \to \infty$, where $W \in \mathbb{M} \times \mathbb{M}$ and $\mathbf{v} \in \mathbb{R}^M$. The purpose of the proposition is to demonstrate how and why the (normalised) vectors $W^k\mathbf{v}$ tend towards a low-dimensional sub-space of $\mathbb{R}^M$. Though we state and prove the result only in the specific case that the maximum-magnitude eigenvalues are a complex conjugate pair, it generalises naturally to other scenarios, with the size of the sub-space being the number of eigenvalues which are joint maximum magnitude such that their associated eigenvectors are not orthogonal to $\mathbf{v}$ (including the multiplicities of repeated eigenvalues).

Proof of this proposition loosely follows the strategy of the standard method of proving the convergence of the power iteration method to the dominant eigenvector of a matrix (i.e., the case where there is a single eigenvalue of largest magnitude and a single associated eigenvector), which can be found in linear algebra textbooks (for instance [AR10]).

Proposition C.3.1. *Let W be a matrix with a simple spectrum such that there exist exactly two eigenvalues of largest magnitude, with these two eigenvalues forming a complex conjugate pair, and $\mathbf{v}$ be a vector with a non-zero component in the direction of these eigenvectors. There exist real vectors $\mathbf{a}, \mathbf{b}$ such that*

$$\frac{W^k\mathbf{v}}{|\rho(W)^k|} = c_{1,k}\mathbf{a} + c_{2,k}\mathbf{b} + c_{3,k}\mathbf{v}_k,$$

where for every natural number k, $\mathbf{v}_k$ satisfies $\|\mathbf{v}_k\| = 1$ and constants $c_{1,k}$, $c_{2,k}$ and $c_{3,k}$ satisfy $\max(|c_{1,k}|, |c_{2,k}|) \geq D_1$ and $c_{3,k} < D_2 \cdot \gamma^k$, for constants $D_1, D_2 > 0$ and $0 < \gamma < 1$.

Proof. Let λ_1, λ_2 be the two eigenvalues of largest magnitude and $\mathbf{p}_1, \mathbf{p}_2$ be their corresponding eigenvectors, and label the other eigenvalues $\lambda_3, \cdots, \lambda_M$ in order of decreasing magnitude. We can decompose $\mathbf{v}$ as $\mathbf{v} = c_1\mathbf{p}_1 + c_2\mathbf{p}_2 + \cdots + c_M\mathbf{p}_M$ for some $c_1, \ldots, c_M$, with $c_1, c_2 \neq 0$. We can write $\mathbf{p}_1 = \mathbf{a} + i\mathbf{b}$ for real vectors $\mathbf{a}$ and $\mathbf{b}$. We have

$$\begin{aligned} W^k\mathbf{v} &= W^k\left(c_1\mathbf{p}_1 + c_2\mathbf{p}_2 + \cdots + c_M\mathbf{p}_M\right) \\ &= \left(c_1 \cdot \lambda_1^k\mathbf{p}_1 + c_2 \cdot \lambda_2^k\mathbf{p}_2 + \cdots + c_M \cdot \lambda_M^k\mathbf{p}_M\right) \end{aligned}$$

Dividing through by $\left|\rho(W)^k\right|$ gives

$$\begin{aligned}\frac{W^k\mathbf{v}}{|\rho(W)^k|} &= \frac{\left(c_1 \cdot \lambda_1^k\mathbf{p}_1 + c_2 \cdot \lambda_2^k\mathbf{p}_2 + \cdots + c_M \cdot \lambda_M^k\mathbf{p}_M\right)}{|\rho(W)^k|} \\ &= c_1 \cdot \frac{\lambda_1^k}{|\rho(W)^k|}\mathbf{p}_1 + c_2 \cdot \frac{\lambda_2^k}{|\rho(W)^k|}\mathbf{p}_2 + \cdots + c_M \cdot \frac{\lambda_M^k}{|\rho(W)^k|}\mathbf{p}_M \\ &= c_1 \cdot \left(\frac{\lambda_1}{|\rho(W)|}\right)^k \mathbf{p}_1 + c_2 \cdot \left(\frac{\lambda_2}{|\rho(W)|}\right)^k \mathbf{p}_2 + \cdots + c_M \cdot \left(\frac{\lambda_M}{|\rho(W)|}\right)^k \mathbf{p}_M.\end{aligned}$$

Define $\mathbf{v}_k = \frac{c_3\cdot\left(\frac{\lambda_3}{|\rho(W)|}\right)^k\mathbf{p}_3+\cdots+c_M\left(\frac{\lambda_M}{|\rho(W)|}\right)^k\mathbf{p}_3}{\left\|c_3\cdot\left(\frac{\lambda_3}{|\rho(W)|}\right)^k\mathbf{p}_3+\cdots+c_M\left(\frac{\lambda_M}{|\rho(W)|}\right)^k\mathbf{p}_M\right\|}$. By the triangle inequality, we have

$$\begin{aligned}&\left\|c_3 \cdot \left(\frac{\lambda_3}{|\rho(W)|}\right)^k \mathbf{p}_3 + \cdots + c_M \left(\frac{\lambda_M}{|\rho(W)|}\right)^k \mathbf{p}_M\right\| \\ &\qquad \le \left\|c_3 \cdot \left(\frac{\lambda_3}{|\rho(W)|}\right)^k \mathbf{p}_3\right\| + \cdots + \left\|c_M \left(\frac{\lambda_M}{|\rho(W)|}\right)^k \mathbf{p}_M\right\| \\ &\qquad = \left|c_3 \cdot \left(\frac{\lambda_3}{|\rho(W)|}\right)^k\right| + \cdots + \left|c_M \left(\frac{\lambda_M}{|\rho(W)|}\right)^k\right| \\ &\qquad < M \cdot \max(|c_3|, \ldots, |c_M|) \cdot \gamma^k,\end{aligned}$$

where γ satisfies $\frac{|\lambda_3|}{\rho(W)} < \gamma < \rho(W)$. Setting $D_2 = M \cdot \max(|c_3|, \ldots, |c_M|)$ gives a bound for the constants $c_{3,k}$.

Next, we note that since λ_1 and λ_2 are a complex conjugate pair, it is also true that $\mathbf{p}_1$ and $\mathbf{p}_2$ are complex conjugates of each other, i.e., $\mathbf{p}_2 = \mathbf{a} - i\mathbf{b}$. From this, we use Euler's formula to write for some θ,

$$\begin{aligned}\frac{1}{|\rho(W)^k|} \cdot \left(c_1 \cdot \lambda_1^k\mathbf{p}_1 + c_2 \cdot \lambda_2^k\mathbf{p}_2\right) =& c_1 \cdot (\cos(k\cdot\theta) + i \cdot \sin(k\cdot\theta))(\mathbf{a} + i\mathbf{b}) \\ &+ c_2 \cdot (\cos(k\cdot -\theta) + i \cdot \sin(k \cdot -\theta))(\mathbf{a} - i\mathbf{b}) \\ =& c_1 \cdot (\cos(k\cdot\theta) + i \cdot \sin(k\cdot\theta))(\mathbf{a} + i\mathbf{b}) \\ &+ c_2 \cdot (\cos(k\cdot\theta) - i \cdot \sin(k\cdot\theta))(\mathbf{a} - i\mathbf{b}) \\ =& \left((c_1 + c_2) \cdot \cos(k\cdot\theta) + i \cdot (c_1 - c_2) \cdot \sin(k\cdot\theta)\right)\mathbf{a} \\ &+ \left(-(c_1 + c_2) \cdot \sin(k\cdot\theta) + i \cdot (c_1 - c_2) \cdot \cos(k\cdot\theta)\right)\mathbf{b}.\end{aligned}$$

We now consider three cases. Firstly, assume $c_1 - c_2 = 0$, then $c_{1,k} = (c_1 + c_2) \cdot$

$\cos(k \cdot \theta)$ and $c_{2,k} = -(c_1 + c_2) \cdot \sin(k \cdot \theta)$. Since for any θ, $\max(|\cos\theta|, |\sin\theta|) \geq \frac{1}{\sqrt{2}}$,[1] we get can set $D_1 = \frac{1}{\sqrt{2}} |(c_1 + c_2)|$. For the second case, assume $c_1 + c_2 = 0$ and we can construct $D_1 = \frac{1}{\sqrt{2}}(c_1 + c_2)$ in a similar manner. Finally, for the case where neither is zero, we get

$$c_{1,k} \overset{\text{def}}{=} (c_1 + c_2) \cdot \cos(k \cdot \theta) + i \cdot (c_1 - c_2) \cdot \sin(k \cdot \theta)$$

and

$$c_{2,k} \overset{\text{def}}{=} -(c_1 + c_2) \cdot \sin(k \cdot \theta) + i \cdot (c_1 - c_2) \cdot \cos(k \cdot \theta).$$

Note that

$$i \cdot c_{2,k} = i \cdot (c_1 + c_2) \cdot \sin(k \cdot \theta) + (c_1 - c_2) \cdot \cos(k \cdot \theta)$$

and therefore

$$\begin{aligned} |c_{1,k} - i \cdot c_{2,k}| &= |2 \cdot c_1 \cdot (\cos(k \cdot \theta) + i \cdot \sin(k \cdot \theta))| \\ &= |2 \cdot c_1| \cdot |(\cos(k \cdot \theta) + i \cdot \sin(k \cdot \theta))| \\ &\geq |2 \cdot c_1| \end{aligned}$$

This implies that at least one of $c_{1,k}$ and $i \cdot c_{2,k}$ must be of magnitude at least $|c_1|$ (since if both had norms smaller than $|c_1|$, by the triangle inequality we would have $|c_{1,k} - i \cdot c_{2,k}| \leq |c_{1,k}| + |i \cdot c_{2,k}| < 2 \cdot |c_1|$)). This gives for all k,

$$\begin{aligned} \max(|c_{1,k}|, |c_{2,k}|) &= \max(|c_{1,k}|, |i \cdot c_{2,k}|) \\ &\geq |c_1|. \end{aligned}$$

Setting $D_1 = |c_1|$ completes the proof (note that we could also have considered $c_{1,k} + i \cdot c_{2,k}$ to get the bound as $|c_2|$). □

[1] This can easily be shown from the identity $cos^2\theta + \sin^2\theta = 1$. For the identity to hold, either $cos^2\theta \geq \frac{1}{2}$ or $\sin^2\theta \geq \frac{1}{2}$, taking the square root gives the result

Appendix D

Supplementary Material for Chapter 6

In this appendix, we provide proof of the various intermediate lemmas stated in the main text of Chapter 6.

D.1 Proofs for Section 6.4

In this section, we provide proof of Lemma 6.4.1. In order to make the proof as clear and straightforward as possible, we first prove the following supplementary lemma.

Lemma D.1.1. *Let W be a matrix such that $\rho(W) = 1$ and let $\mathbf{v}$ be aligned to W with degree $d \geq 1$. Define $\boldsymbol{v} \overset{\text{def}}{=} \mathbf{v}\epsilon$, where ϵ is a random variable with zero mean and finite variance. There exists constants $k_0, C_1, C_2 > 0$ such that for all $k > k_0$*

$$C_1 \cdot k^{2\cdot(d-1)} < \mathbb{E}\left[\left\|W^k \boldsymbol{v}\right\|^2\right] < C_2 \cdot k^{2\cdot(d-1)}.$$

Proof. We can show that the same inequalities hold when using the norm $\|\cdot\|_P$, though not necessarily for the same constants, then use the equivalence of norms[1]. Consider the components of the vector $P^{-1}W^k\boldsymbol{v}$: for $1 \leq i \leq M$, if the eigenvalue associated with the ith component λ_i satisfies $|\lambda_i| < 1$, then with a small amount of manipulation Equation 6.3 tells us that the magnitude of the component in that direction decays asymptotically exponentially with k. For the remaining elements,

[1]We remind the reader that $\|\cdot\|_P$ is defined in Chapter 6 as the norm such that for any $\mathbf{x} \in \mathbb{R}^M$, $\|\mathbf{x}\|_P = \left\|P^{-1}\mathbf{x}\right\|_\infty$

$|\lambda_i| = 1$, and there exists a polynomial $p_i(k)$ such that $\left|(P^{-1}W^k\boldsymbol{v})_i\right| = |p_i(k) \cdot \epsilon|$. Define S as the set of all such i, this gives that for large enough k,

$$\begin{aligned}
\mathbb{E}\left[\left\|W^k\mathbf{v}\epsilon\right\|_P^2\right] &= \mathbb{E}\left[\left(\max_{i\in S}\left(\left(\left|(P^{-1}W^k\boldsymbol{v})\right|\right)_i\right)\right)^2\right] \\
&= \mathbb{E}\left[\left(\max_{i\in S}(|p_i(k)|\cdot|\epsilon|)\right)^2\right] \\
&= \max_{i\in S}\left(|p_i(k)|^2\cdot\mathbb{E}\left[|\epsilon|^2\right]\right)
\end{aligned}$$

Since the largest degree amongst the polynomials $p_i(k)$ is $d-1$, we have that there exist positive constants C_1', C_2' and t_0 such that for all $k > k_0$

$$C_1' \cdot k^{2\cdot(d-1)} \le \mathbb{E}\left[\left\|W^k\boldsymbol{v}\right\|_P^2\right] \le C_2' \cdot k^{2\cdot(d-1)}$$

Using equivalence of norms we can get the desired bounds for the 2-norm. □

With this lemma in place, we are ready to proceed to Lemma 6.4.1.

Lemma 6.4.1. *Let W be a matrix such that $\rho(W) = 1$ and $\mathbf{v}$ a vector which is aligned to W with degree $d \ge 1$. Define $\boldsymbol{v}_k = \mathbf{v}\epsilon_k$, where each ϵ_k is a random variable with zero-mean and the same finite variance. There exist constants $C_1, C_2, t_0 > 0$ such that for all $t > t_0$*

$$C_1 \cdot t^{2\cdot d-1} < \mathbb{E}\left[\sum_{k=0}^{t-1}\left\|W^k\boldsymbol{v}_{t-k}\right\|^2\right] < C_2 \cdot t^{2\cdot d-1}$$

Proof. By Lemma D.1.1 we have that there exists $C_1', C_2', k_0 > 0$ such that for all $k \ge k_0$

$$C_1' \cdot k^{2\cdot(d-1)} < \mathbb{E}\left[\left\|W^k\boldsymbol{v}\right\|^2\right] < C_2' \cdot k^{2\cdot(d-1)}$$

Let $C_0 = \mathbb{E}\left[\sum_{k=0}^{k_0-1}\left\|W^k\boldsymbol{v}_{t-k}\right\|^2\right]$. We can construct a lower bound using that for all $t > 2 \cdot k_0$, we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}_k\right\|^2\right] &= C_0 + \mathbb{E}\left[\sum_{k=k_0}^{t-1} \left\|W^k \boldsymbol{v}_k\right\|^2\right] > C_0 + \sum_{k=k_0}^{t-1} C_1' \cdot k^{2\cdot(d-1)} \\
&> \sum_{k=k_0}^{t-1} C_1' \cdot k^{2\cdot(d-1)} \geq \sum_{k=\lceil \frac{t}{2} \rceil}^{t-1} C_1' \cdot k^{2\cdot(d-1)} \\
&\geq \sum_{k=\lceil \frac{t}{2} \rceil}^{t-1} C_1' \cdot \left(\frac{t}{2}\right)^{2\cdot(d-1)} = \sum_{k=\lceil \frac{t}{2} \rceil}^{t-1} \frac{C_1'}{2^{2\cdot(d-1)}} \cdot t^{2\cdot(d-1)} \\
&= \left(t - \lceil \frac{t}{2} \rceil\right) \cdot \frac{C_1'}{2^{2\cdot(d-1)}} \cdot t^{2\cdot(d-1)} \\
&\geq \left(t - \frac{t}{2} - 1\right) \cdot \frac{C_1'}{2^{2\cdot(d-1)}} \cdot t^{2\cdot(d-1)} \\
&= \left(\frac{t}{2} - 1\right) \cdot \frac{C_1'}{2^{2\cdot(d-1)}} \cdot t^{2\cdot(d-1)} \\
&= \frac{C_1'}{2^{2\cdot d-1}} \cdot \left(t^{2\cdot d-1} + 2 \cdot t^{2\cdot d-2}\right) > C_1 \cdot t^{2\cdot d-1},
\end{aligned}$$

for some constant C_1 and sufficiently large t.

For the upper bound, we use the other inequality in Lemma D.1.1 to get

$$\begin{aligned}
\mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \mathbf{v}\right\|^2\right] &= C_0 + \mathbb{E}\left[\sum_{k=k_0}^{t-1} \left\|W^k \mathbf{v}\right\|^2\right] \\
&< C_0 + \sum_{k=k_0}^{t-1} C_2' \cdot k^{2\cdot(d-1)} \\
&< C_0 + \sum_{k=k_0}^{t-1} C_2' \cdot t^{2\cdot(d-1)} \\
&< C_0 + (t - 1 - k_0 + 1) \cdot C_2' \cdot t^{2\cdot(d-1)} \\
&< C_0 + t \cdot C_2' \cdot t^{2\cdot(d-1)} \\
&= C_0 + C_2' \cdot t^{2\cdot d-1} \\
&< (C_0 + C_2') \cdot t^{2\cdot d-1},
\end{aligned}$$

therefore setting $C_2 \overset{\text{def}}{=} C_0 + C_2'$ completes the proof. $\square$

D.2 Proofs for Section 6.5

In this appendix, we provide proof of Lemma 6.5.1. Mirroring the structure of the previous section, we begin with the following lemma.

Lemma D.2.1. *Let W be a matrix such that $\rho(W) > 1$. Define $\boldsymbol{v} = \mathbf{v}\epsilon$, where ϵ is a random variable, with zero mean and non-zero variance, and $\mathbf{v}$ is aligned to W with degree at least one. For any $\epsilon > 0$, there exist constants $C_1, C_2, k_0 > 0$, such that for all $k > k_0$,*

$$C_1 \cdot \rho(W)^{2\cdot k} < \mathbb{E}\left[\left\|W^k \boldsymbol{v}\right\|^2\right] < C_2 \cdot (\rho(W) + \varepsilon)^{2\cdot k}.$$

Proof. By equivalence of norms it suffices to consider whether the inequalities hold for $\mathbb{E}\left\|W^k\boldsymbol{v}\right\|_P$ (albeit for different constants C_1' and C_2'). By Equation 6.3, we have the magnitude of the ith component can be written as

$$\left|(P^{-1}W^k\boldsymbol{v})_i\right| = \left|\lambda^k \cdot p_i(k) \cdot \epsilon\right|,$$

where $p_i(k)$ is a polynomial in k. We can write

$$\begin{aligned}
\mathbb{E}\left[\left\|W^k\boldsymbol{v}\right\|_P^2\right] &= \mathbb{E}\left[\left(\max_{1\le i\le M}\left|(P^{-1}W^k\boldsymbol{v})_i\right|\right)^2\right] \\
&= \mathbb{E}\left[\left(\max_{1\le i\le M}\left|\lambda_i^k \cdot p_i(k)\right|\left|\epsilon\right|\right)^2\right] \\
&= \max_{1\le i\le M}\left|\lambda^k \cdot p_i(k)^2\right| \cdot \mathbb{E}\left[\left|\epsilon\right|^2\right]. \qquad \text{(D.1)}
\end{aligned}$$

For large enough k, the maximum will correspond to an i satisfying $|\lambda_i| = \rho(W)$, since if $|\lambda_i| < \rho(W)$, asymptotically $\left|(\lambda_i^k \cdot p_i(k))^2\right| < \left|\rho(W)^{2\cdot k}\right|$. This means that for large enough k, Equation D.1 yields

$$\mathbb{E}\left[\left\|W^k\boldsymbol{v}\right\|_P^2\right] > C_1 \cdot \rho(W)^{2k} \cdot \mathbb{E}\left[|\epsilon|^2\right],$$

for some constant $C_1' > 0$, giving us a lower bound. We also have that for any $\varepsilon' > 0$, for all for all sufficiently large k,

$$\left|p_i(k)^2\right| < \left|(1+\varepsilon')^{2\cdot k}\right|,$$

Setting $\epsilon' = \frac{\epsilon}{\lambda}$, we get that for all sufficiently large k

$$\begin{aligned} \left|\lambda^{2\cdot k} \cdot p_i(k)^2\right| &< \left|\lambda^{2\cdot k} \cdot (1+\epsilon')^{2\cdot k}\right| \\ &= \left|(\lambda+\varepsilon)^{2\cdot k}\right| = (\rho(W)+\varepsilon)^{2\cdot k}. \end{aligned} \tag{D.2}$$

Combining this with Equation D.1, we have for sufficiently large k,

$$\mathbb{E}\left[\left\|W^k \boldsymbol{v}\right\|_P^2\right] < C_2' \cdot (\rho(W)+\epsilon)^{2\cdot k} \cdot \mathbb{E}\left[|\epsilon|\right]^2. \tag{D.3}$$

From inequalities D.2 and D.3, we get the desired result by the equivalence of norms. □

With this lemma in hand, we proceed to the proof of Lemma 6.5.1

Lemma 6.5.1. *Let W be a matrix such that $\rho(W) > 1$, and $\mathbf{v}$ be a vector which is aligned with W with degree $d \geq 1$. Define $\boldsymbol{v}_k = \mathbf{v}\epsilon_k$, where each ϵ_k is a random variable with zero mean and the same non-zero finite variance. For any $\varepsilon > 0$, there exist constants $t_0 > 0$, $C_1 > 0$ and $C_2 > 0$ such that for all $t \geq t_0$*

$$C_1 \cdot \rho(W)^{2\cdot k} < \mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] < C_2 \cdot (\rho(W)+\varepsilon)^{2\cdot k}$$

Proof. By Lemma D.2.1 we have that there exists $C_1', C_2', k_0 > 0$ such that for all $k \geq k_0$

$$C_1' \cdot \rho(W)^{2\cdot k} < \mathbb{E}\left[\left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] < C_2' \cdot (\rho(W)+\varepsilon)^{2\cdot k}.$$

Let $C_0 = \mathbb{E}\left[\sum_{k=0}^{k_0-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|\right]$. We have

$$\begin{aligned} \mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] &= C_0 + \mathbb{E}\left[\sum_{k=k_0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] \\ &> C_0 + \sum_{k=k_0}^{t-1} C_1' \cdot \rho(W)^{2\cdot k} \\ &\geq C_0 + C_1' \cdot \rho(W)^{2\cdot(t-1)} \\ &= C_0 + C_1' \cdot \rho(W)^{-2} \cdot \rho(W)^{2\cdot t} \\ &> C_1' \cdot \rho(W)^{-2} \cdot \rho(W)^{2\cdot t} \end{aligned}$$

Therefore for if we define $C_1 = C_1' \cdot \rho(W)^{-2}$, for $t > k_0$, we have the lower bound. For the upper bound, we have for some constant $C_3 > 0$ and ε' such that $0 < \varepsilon' < \varepsilon$

$$\begin{aligned}
\mathbb{E}\left[\sum_{k=0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] &= C_0 + \mathbb{E}\left[\sum_{k=k_0}^{t-1} \left\|W^k \boldsymbol{v}_{t-k}\right\|^2\right] \\
&< C_0 + \sum_{k=k_0}^{t-1} C_2' \cdot (\rho(W) + \varepsilon')^{2\cdot k} \\
&\leq C_0 + \sum_{k=k_0}^{t-1} C_2' \cdot (\rho(W) + \varepsilon')^{2\cdot (t-1)} \\
&\leq C_0 + (t - 1 - k_0 + 1) \cdot C_2' \cdot (\rho(W) + \varepsilon')^{2\cdot (t-1)} \\
&= C_0 + (t - k_0) \cdot C_2' \cdot (\rho(W) + \varepsilon')^{2\cdot (t-1)} \\
&< C_0 + C_2' \cdot C_3 \cdot (\rho(W) + \varepsilon')^{2\cdot (t-1)} \\
&= C_0 + C_2' \cdot C_3 \cdot (\rho(W) + \varepsilon)^{-2} \cdot (\rho(W) + \varepsilon)^{2\cdot t} \\
&< (C_0 + C_2' \cdot C_3 \cdot (\rho(W) + \varepsilon)^{-2}) \cdot (\rho(W) + \varepsilon)^{2\cdot t}
\end{aligned}$$

Defining $C_2 = (C_0 + C_2' \cdot C_3 \cdot (\rho(W) + \varepsilon)^{-2})$ and setting $t_0 = k_0$ completes the proof. □

The next two proofs are mostly trivial modifications of proofs from Section 6.4. As such, we provide abridged versions of the proof, and refer readers to the analogous proof in that section for additional detail of intermediate steps.

Corollary 6.5.1. *Let $\{\boldsymbol{u}_t\}_{t=1}^{\infty}$ be a sequence of random vectors in $\mathbb{R}^D$, with the entries of all vectors being zero-mean and a common non-zero finite variance. For the ESN (W, V) driven by that sequence, if $\rho(W) > 1$ and there exists at least one column of V aligned with W with degree at least one, then for all $\varepsilon > 0$ there exist $C_1 > 0$, $C_2 > 0$, $t_0 > 0$ such that for all $t > t_0$,*

$$C_1 \cdot \rho(W)^t < \mathbb{E}\left[\|\mathbf{x}_t\|\right] < C_2 \cdot \rho(W + \varepsilon)^t.$$

Proof. First, note that

$$\boldsymbol{x}_t = \sum_{i=1}^{D} \sum_{k=0}^{t-1} W^k V_{\cdot,i} (\boldsymbol{u}_{t-k})_i,$$

where $V_{\cdot,i}$ is the ith column of V. Define

$$\boldsymbol{x}_t^{(i)} \overset{\text{def}}{=} \sum_{k=0}^{t-1} W^k V_{\cdot,i} (\boldsymbol{u}_{t-k})_i,$$

We can write $\boldsymbol{x}_t = \sum_{i=1}^{D} \boldsymbol{x}_t^{(i)}$, giving that $\boldsymbol{x}_t$ is the sum of independent random vectors. The upper bound can easily be obtained by splitting the norm of this sum into a sum of norms using the triangle inequality. That is to say, we have for any $\varepsilon < 0$, there exist $C_2, t_0 > 0$ such that for all $t > t_0$

$$\begin{aligned} \mathbb{E}\left[\|\boldsymbol{x}_t\|\right] &= \mathbb{E}\left[\left\|\sum_{i=1}^{D} \boldsymbol{x}_t^{(i)}\right\|\right] \\ &\leq \mathbb{E}\left[\sum_{i=1}^{D} \left\|\boldsymbol{x}_t^{(i)}\right\|\right] \\ &\leq D \cdot C_2 \cdot (\rho(W) + \epsilon)^t \end{aligned}$$

Using the same argument as in Corollary 6.4.1, we have

$$\mathbb{E}\,\|\boldsymbol{x}_t\| \geq \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{\mathbb{E}\left[\sum_{i=1}^{D} \left\|\boldsymbol{x}_t^{(i)}\right\|^2\right]}$$

Combining this with Lemma 6.5.1 yields

$$\begin{aligned} \mathbb{E}\,\|\boldsymbol{x}_t\| &\geq \frac{1}{2 \cdot C_{1,2}} \cdot \sqrt{C_1' \cdot D \cdot (\rho(W))^{2 \cdot t}} \\ &= \frac{\sqrt{C_1' \cdot D}}{2 \cdot C_{1,2}} \cdot (\rho(W))^t. \end{aligned}$$

Setting $C_1 = \frac{\sqrt{C_1' \cdot D}}{2 \cdot C_{1,2}}$ completes the proof. $\square$

Corollary 6.5.2. *Let (W, V) be an ESN satisfying $\rho(W) > 1$. Let $d \geq 1$ be the greatest degree of alignment of columns of V with W. Let $\mathbf{x}_t$ be the state of the ESN (W, V) driven by the input sequence $\{\boldsymbol{u}_t\}_{t=1}^{\infty}$ where for each t, $\boldsymbol{u}_t = \mathbf{u}_t + \boldsymbol{\epsilon}_t$ with each $\mathbf{u}_t$ as some vector in $\mathbb{R}^D$ and each $\boldsymbol{\epsilon}_t$ as a vector of the same size whose entries are zero-mean i.i.d. random variables, with the entries of all $\boldsymbol{\epsilon}_t$ having the same non-zero variance. For all $\varepsilon > 0$, there exist $C_1, C_2, t_0 > 0$ such that for all*

$t > t_0$,

$$C_1 \cdot \rho(W)^t < \mathbb{E}\left[\|\boldsymbol{x}_t\|\right] < \left\| \sum_{k=0}^{t-1} W^k V \mathbf{u}_t \right\| + C_2 \cdot (\rho(W) + \varepsilon)^t.$$

Proof. For the upper bound, we have

$$\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] \leq \mathbb{E}\left[\left\| \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} \right\| + \left\| \sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k} \right\|\right]$$

Combining this with Theorem 6.5.1 on the above gives for sufficiently large t

$$\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] < \left\| \sum_{k=0}^{t-1} W^k V \boldsymbol{u}_t \right\| + C_2 \cdot (\rho(W) + \varepsilon)^t$$

For the lower bound, we use Theorem 6.4.1 to say that for some constants C_3 and all $t_0 > t$

$$C_3 \cdot \rho(W)^t < \left\| \sum_{t=0}^{k-1} W^k V \boldsymbol{\epsilon}_{t-k} \right\|. \tag{D.4}$$

Now choose $C_4 < C_3$ and define $\mathbf{x}'_t = \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k}$. For all $t > t_0$, if $\|\mathbf{x}'_t\| > C_4 \cdot \rho(W)^t$, the convexity of the norm along with Jensen's inequality gives

$$\begin{aligned} \mathbb{E}\left[\|\mathbf{x}_t\|\right] &= \mathbb{E}\left[\left\| \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} + \sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k} \right\|\right] \\ &\geq \left\| \sum_{k=0}^{t-1} W^k V \mathbf{u}_{t-k} \right\| \\ &> C_4 \cdot \rho(W)^t \end{aligned}$$

Otherwise, we have $\|\mathbf{x}'_t\| \leq C_4 \cdot \rho(W)^t$ and we have,

$$\mathbb{E}\left[\|\boldsymbol{x}_t\|\right] \geq \left| \mathbb{E}\left[\left\| \sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k} \right\|\right] - \|\mathbf{x}'_t\| \right|.$$

We can now use Equation D.4 to get

$$\begin{aligned}\left|\mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right] - \|\mathbf{x}'_t\|\right| &\geq \left|\underbrace{\mathbb{E}\left[\left\|\sum_{k=0}^{t-1} W^k V \boldsymbol{\epsilon}_{t-k}\right\|\right]}_{>C_3\cdot\rho(W)^t} - \underbrace{\|\mathbf{x}'_t\|}_{\leq C_4\cdot\rho(W)^t}\right| \\ &> |(C_3 - C_4)| \cdot \rho(W)^t \\ &= C_1 \cdot \rho(W)^t\end{aligned}$$

Defining $C_1 \overset{\text{def}}{=} \min(C_4, C_3 - C_4)$ completes the proof. $\square$