Adaptive Functional Modeling of Neural Activity:
A Reservoir Computing Approach to Neuronal Cultures

Tayfun GÜREL

Dissertation

zur Erlangung des Doktorgrades der Technischen Fakultät
der Albert-Ludwigs-Universität Freiburg im Breisgau
August 2010

Betreuer:
Prof. Dr. Ulrich Egert
Dekan:
Prof. Dr. Bernd Becker

Prüfungskomission:
Prof. Dr. Wolfram Burgard (Vorsitz)
Prof. Dr. Luc De Raedt (Beisitz)
Prof. Dr. Ulrich Egert (Betreuer)
Prof. Dr. Stefan Rotter (Gutachter)

Datum der Disputation: 4. November 2010
Dedicated to the Memory of My Mother
Acknowledgments

This thesis would not have been written without the productive environment composed of many colleagues.

Firstly, I would like to thank to my thesis supervisors Ulrich Egert, Stefan Rotter and Luc De Raedt for their advices and suggestions that made this thesis possible. I am also grateful to Herbert Jaeger, whose lab I could visit in 2007 in order to gain theoretical and practical aspects of Echo State Networks and with whom I had fruitful discussions.

I had a close working environment with Kristian Kersting, Samora Okujeni, Steffen Kandler, Oliver Weihberger, Sarah Jarvis during the work towards this thesis. I appreciate this productive working environment. I also appreciate the contributions of Ad Aertsen, Birgit Kriener, Abigail Morrison, Bernd Wiebelt, Melisa Mustafovic, Susanne Bourjoillat, Florence Dancoisne, Anne Rottler, Gunnar Grah, Simone Cardoso, Janina Kirsch for their scientific opinions, technical and administrative assistance, which have been very helpful throughout my doctoral work.

I want to thank to my friends and colleagues from Bernstein Center Freiburg, Institute for Informatics and IMTEK for providing a nice atmosphere. These include Heidi Teppola, Patrick Dini, Yuri Zaytsev, Sebastian Reinartz, Claudia Böhm, Nila Möhing, Stefano Cardanobile, Bjoern Bringmann, Bernd Gutman, Albrecht Zimmermann, Jannis Vlachos, Niels Landwehr, Arvind Kumar and the other BCF members.

I express my special gratitude to my family including my mother, father and brother. Without them I would not even start doing this work.

I acknowledge Bernstein Center funders, especially Bundesministerium für Bildung und Forschung for making this work possible.
Contributions from my colleagues

Computational methods developed in the scope of this thesis are my own work. Without experimental work that led to data harvesting, however, these methods would be of no use. Here, I appreciate three names, whose experimental work produced the data that is used to test the methods in this work.

Steffen Kandler’s experimental work included culture preparation and electrophysiological recordings in chapter 4 and partially for chapter 6. Oliver Weihberger’s experimental work included a part of the electrophysiological recordings in chapters 6 and 7. Samora Okujeni’s experimental work included culture preparation and electrophysiological recordings in chapter 7.
Publications of the author during the thesis work

Published Journal Articles


Journal Articles in preparation

Book Chapters

Conference Articles

Note that abstracts, posters and talks are not listed above.
Biologische neuronale Netze (BNN) sind sehr reich und komplex bezüglich ihrer Struktur und ihren zeitlich-räumlichen Aktivitätsmustern. Diese Komplexität verhindert die unmittelbare Abbildung ihrer anatomischen und biophysikalischen Eigenschaften auf ihre dynamische elektrische Aktivität. Es ist daher nicht ohne weiteres möglich die funktionelle Relevanz ihrer Struktur für die Aktivitätsdynamik analytisch herauszufinden. Der Schwerpunkt dieser Doktorarbeit ist die Entwicklung von Adaptive-Computing-Tools, d.h. Lernalgorithmen, zur funktionellen Modellierung der BNN.

Kortikale Zellen mancher neugeborenen Säugetiere, in der vorliegenden Arbeit von Ratten, können aus dem Gehirn isoliert werden und in Zellkulturen neuronale Netze ausbilden, die als geschlossene Systeme betrachtet werden können. Zur funktionellen Modellierung von BNN verwenden wir also Kulturen von kortikalen Neuronen als Bezugsprofile der BNN. Die Anatomie des Gehirns ist in den neuronalen Kulturen nicht erhalten, infolgedessen spiegelt deren Konnektivität nicht die Struktur im Gehirn wider, die grundlegenden Eigenschaften der Neurone und die Bedingungen für die Netzwerkbildung bleiben jedoch erhalten. Da diese BNNs keine inhärente Funktion haben, erfolgte die funktionelle Modellierung von folgenden zwei Aufgabenstellungen aus:


ABSTRACT

Biological neural networks (BNN) are very rich and complex in terms of structure and spatiotemporal activity patterns. This complexity does not allow to directly relate their anatomical and biophysical properties to the dynamics of their electrical activity. Functional relevance of their structure and dynamics is also difficult to track analytically. The focus of this thesis is the development of adaptive computing tools, i.e. learning algorithms, for functional modeling of BNNs.

Mammalian cortical cells can be dissociated from the brain and regrown outside of the body. They form \textit{in vitro} closed system networks. To study functional modeling of BNNs, we utilize dissociated cultures of cortical neurons as reference BNNs. The anatomy in the brain is not anymore conserved in cultured networks, hence their connectivity does not reflect the structure in the brain. Neuronal cultures also do not have natural functions. Thus, we approach to functional modeling in neuronal cultures by investigating two problem settings:

1) We first regard the input-output relation of a BNN as a very detailed characterization of its function and model its response streams to input streams. We describe this approach as functional identification of a BNN, i.e. building an artificial system that is functionally equivalent to the reference BNN.

2) We regard the self-organized temporal activity patterns in cultures, i.e. dynamic attractors, as characteristics of their functions, as it has been argued that spatiotemporal patterns and dynamic attractors in \textit{in vivo} brains reflect functionally relevant brain states. Here, we model the temporal activity patterns in the cultures based on a single cue signal that reflects the initial onset of the temporal pattern. Based on a cue signal, the model simulates the temporal pattern that the neuronal culture undergoes.

We employ reservoir computing as a framework, which is argued to be a generic network model for cortical information processing. More specifically, we use Echo State Networks (ESN) of leaky integrator neurons. There have been efficient learning algorithms for ESNs that work on continuous valued time varying signals. For task 1, we take a point process framework rather than a continuous valued signal approach. Using feed-forward and recurrent reservoirs, we adapt the existing ESNs algorithms for point process data and develop their extensions for reservoir adaptation. The proposed algorithm learns predictive models of input-output relations. Receiver Operating Characteristic (ROC) curve and cross correlation analyses indicate that our ESN models can imitate the response signal of a reference biological network for random pulse train inputs. Moreover, point process based reservoir adaptation improves the performance of an ESN over readout-only training methods in many cases. This also holds for adaptive feed-forward reservoirs, which have no recurrent dynamics. Indeed, we show that feed-forward reservoirs perform significantly better than recurrent reservoirs for some of the subtasks.

For task 2, we employ an Echo State Network (ESN) with leaky integrator neurons...
as a modeling tool. A simple readout training approach yields a predictive model of the temporal activity pattern in the global firing rate. The initial state at the pattern onset is used as a cue and the attractor is simulated by the model. The simulated attractor displays a high correlation with the actual one observed in the culture. High correlation measures and successful prediction of attractor-types indicate a good modeling quality. The developed model can also be used for visualizing the underlying structure in the recorded raw data. Investigating the model dynamics reveals multiple types of attractor states in the system.

In addition to functional modeling of neuronal cultures, we show that adaptive computing tools can be used for structural mining of BNNs. We propose the NeuronRank algorithm, which extracts structural features from the connectivity of a BNNs. These structural features are eventually used for predictors of the activity measures in the network. Inspired by the PageRank and the Hubs & Authorities algorithms for networked data, the NeuronRank algorithm assigns a source value and a sink value to each neuron in the network. Distribution over source and sink values can successfully predict average firing rates in the network, as well as the firing rate of output neurons reflecting the network population activity.
### List of abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMI</td>
<td>Brain Machine Interfaces</td>
</tr>
<tr>
<td>BN</td>
<td>Bayesian Network</td>
</tr>
<tr>
<td>BNN</td>
<td>Biological Neural Network</td>
</tr>
<tr>
<td>cc</td>
<td>clustering coefficient</td>
</tr>
<tr>
<td>ESN</td>
<td>Echo State Network</td>
</tr>
<tr>
<td>FF-ESN</td>
<td>Feedforward Echo State Network</td>
</tr>
<tr>
<td>FF-fxd</td>
<td>Feedforward fixed</td>
</tr>
<tr>
<td>IAF</td>
<td>Integrate-and-Fire</td>
</tr>
<tr>
<td>LI-ESN</td>
<td>Leaky-Integrator Echo State Network</td>
</tr>
<tr>
<td>LSM</td>
<td>Liquid State Machine</td>
</tr>
<tr>
<td>MEA</td>
<td>Microelectrode Array</td>
</tr>
<tr>
<td>R-adp</td>
<td>Recurrent adaptive</td>
</tr>
<tr>
<td>R-fxd</td>
<td>Recurrent fixed</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating Characteristic</td>
</tr>
<tr>
<td>RTRL</td>
<td>Real Time Recurrent Learning</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
</tbody>
</table>
## Contents

1 Introduction ........................................... 1
   1.1 Neurons and Synapses ................................. 2
   1.2 Cortical Activity and Its Variability ................. 3
   1.3 Necessity for Adaptive Methods in Modeling Neural Activity .... 4
   1.4 The "Model" Cortex .................................. 5
   1.5 The Modeling Tool .................................... 6
   1.6 Motivation and Scope of This Thesis .................. 6
      1.6.1 Findings and contributions ....................... 10
      1.6.2 How is the rest of this thesis organized? .......... 11

2 Cortical Cell Cultures ................................. 13
   2.1 Introduction ........................................... 13
   2.2 Culture Preparation ................................. 15
   2.3 Electrophysiology ..................................... 15
   2.4 Activity and Dynamics ................................ 16
   2.5 Pharmacological Treatments of Neuronal Cultures ......... 19
   2.6 Induced Changes and Learning in Neuronal Cultures ....... 19
   2.7 Artificial Functions for Neuronal Cultures ............ 20

3 Point Process Modeling of Neural Systems ............... 23
   3.1 Introduction ........................................... 23
   3.2 Point Process Log-Likelihood .......................... 24
   3.3 Generalized Linear Models and Point Process Modeling of Neural Systems .... 25
   3.4 Adaptive Point Process Modeling ....................... 25
      3.4.1 Gradient Descent Learning ......................... 26
      3.4.2 Maximum Likelihood Approaches .................... 26
      3.4.3 Adaptive Point Process Filtering ................... 26

4 Spike Prediction in Neuronal Cultures .................. 29
   4.1 Introduction ........................................... 29
   4.2 The Model ............................................. 32
      4.2.1 Modeling the spike activity as point processes ........ 33
   4.3 The Learning Algorithm ................................ 34
   4.4 Experimental Results .................................. 34
      4.4.1 Evaluation of the model with binary prediction ....... 38
      4.4.2 Implications of the results ......................... 39
   4.5 Conclusion ............................................. 40
## 5 Reservoir Computing

5.1 Introduction ................................................. 41

5.2 Echo State Networks ........................................... 42
   5.2.1 Description of Echo States .............................. 43
   5.2.2 Dynamics .............................................. 43
   5.2.3 Echo State Property .................................... 44
   5.2.4 Existence of Echo States .............................. 44
   5.2.5 Training Echo State Networks ......................... 45
   5.2.6 Echo State Networks with Leaky Integrators Neurons ... 46
   5.2.7 Feed-Forward Echo State Networks .................... 46

5.3 Liquid State Machines ....................................... 47
   5.3.1 Dynamics .............................................. 47
   5.3.2 Computational Power of Liquid State Machines ........ 48
   5.3.3 Further remarks on LSMs .............................. 48

## 6 Functional Modeling of Biological Neural Networks using Reservoir Adaptation for Point Processes

6.1 Introduction ................................................. 51
   6.1.1 Dissociated cortical cultures as reference biological neural networks .......................... 52
   6.1.2 Problem Statement ..................................... 53

6.2 The Echo State Network Model .............................. 55
   6.2.1 ESN with Leaky Integrators .......................... 55
   6.2.2 Feed-Forward Echo State Networks ................... 57
   6.2.3 ESN Learning ........................................ 57

6.3 Point Process Modeling of Spike Data ........................ 58

6.4 Point Process Modeling with ESN and Log Likelihood Propagation ....................... 59
   6.4.1 Reservoir Adaptation ................................ 59
   6.4.2 Existence of Local Maxima and Confidence Intervals .................. 61

6.5 Evaluation of the Learned Models ................................................. 62

6.6 Experimental Results ........................................ 62
   6.6.1 Simulations of Random Cortical Networks ............. 62
   6.6.2 Prediction of Spontaneous Events in Neuronal Cultures ... 73
   6.6.3 Next-Event Prediction in Neuronal Cultures ........... 74

6.7 Conclusion ................................................. 80

## 7 Modeling Dynamic Attractor States of Biological Neural Networks using Reservoir Computing

7.1 Introduction ................................................. 81

7.2 The ESN Model ............................................. 82
   7.2.1 ESN Learning ........................................ 82

7.3 Methods for Modeling and Evaluation of the Learned Models .......................... 84
   7.3.1 Modeling ............................................. 84
8 An Adaptive Computing Method for Mining the Structure of Biological Neural Networks: NeuronRank

8.1 Introduction
8.1.1 Link mining and networked data
8.1.2 Neuronal networks of the mammalian cortex
8.1.3 Motivation and scope of this chapter

8.2 Overview of the method

8.3 The network models
8.3.1 Neuron model
8.3.2 Types of networks

8.4 Activity-related features

8.5 The NeuronRank Algorithm

8.6 Structural features
8.6.1 NeuronRank features
8.6.2 Network motifs
8.6.3 Cluster coefficient

8.7 Experimental results

8.8 Conclusions

9 Discussion

A

B

Bibliography
Chapter 1

Introduction

Contents

1.1 Neurons and Synapses ................................................. 2
1.2 Cortical Activity and Its Variability ................................. 3
1.3 Necessity for Adaptive Methods in Modeling Neural Activity ........ 4
1.4 The "Model" Cortex ...................................................... 5
1.5 The Modeling Tool ....................................................... 6
1.6 Motivation and Scope of This Thesis ................................. 6
1.6.1 Findings and contributions ........................................ 10
1.6.2 How is the rest of this thesis organized? ......................... 11

Human brain is the most complex object known to humanity. From medulla oblongata in the bottom to the cerebral cortex on the top, various anatomically distinct structures can be identified and linked to different stages of evolution. Despite its well known anatomy, far less is understood about its physiology. The major objective of neurosciences is to understand how the brain represents, processes, and conveys information. The enthusiasm for understanding the human brain has lead to many biologically founded theories to explain the functioning of the brain, including cell assemblies [Brown 2003b, Hebb 1949] born in psychology, liquid state machines [Maass 2002] emerged in computer science and synfire chains [Abeles 1993] developed in neurophysiology. Many theories have been inspiring the spirit of research in the area, however, the unknown about the brains working mechanisms by far exceeds the known facts about it. An exceptionally interesting structure is the mammalian cerebral cortex, which is the locus of many brain functions related to sensory perception, voluntary movement control and ‘higher’ cognitive processes, such as vision, hearing, speech, planning, problem solving and thinking. The neocortex, a part of the cerebral cortex, is the outer sheet on top of the brain’s cerebral hemispheres. Having a thickness of 2-4 mm, the neocortex contains roughly $10^5$ neurons/mm$^3$, reaching up to around $2.5 \times 10^{10}$ neurons totally [Nolte 1993]. The computational power of the neocortex is not attributed to the number of neurons only, rather to its extensive number of synapses. A local circuit of 1 mm$^3$ neocortex has around $10^4$ synapses per neuron [Braitenberg 1998]. In the rest of this thesis, when we mention findings about the cortex, we refer to those about the neocortex.
Figure 1.1: Equivalent circuit for the integrate-and-fire neuron. In this model, spikes are generated by a simple threshold mechanism. The voltage between the lower and the upper nodes corresponds the membrane potential.

1.1 Neurons and Synapses

Being basic building blocks of the brain, neurons are generators of action potentials [Hodgkin 1952, Kandel 2000], which can be thought of as uniform electrical pulses. Action potentials, commonly referred as spikes, are believed to be elemental quanta of information processing in the brain. Neurons generate spikes in their cell body in response to the integrated incoming spikes from other neurons. The incoming spikes are received in form of conductances and associated current inflow. They are integrated into an dynamic transmembrane potential through the dendritic tree of the neuron. A simple mathematical model of a neuron is the integrate-and-fire model, where incoming spike integration is linear,

$$\tau \dot{V}(t) = -V(t) + RI(t).$$

Here, $V(t)$ denotes the transmembrane potential of the neuron. $\tau$ is the membrane time constant and $I(t)$ is the current reflecting incoming spikes to the neuron. $R$ can be thought as the transmembrane resistance. The equivalent circuit describing the neuronal membrane is shown in Figure 1.1.

Current inflow is induced by the synapses, which transfer the incoming spikes from other neurons,

$$RI(t) = \tau \sum_i J_i \sum_k \delta(t - t_i^k - D),$$

where $\tau$ is the membrane time constant, $J_i$ is the efficacy of the synapse with neuron $i$, $t_i^k$ are the spike times of neuron $i$, $D$ is a fixed transmission delay, and $\delta$ is the Dirac delta-function. Integrate-and-fire neuron models the spike generation by a hard threshold nonlinearity. If the membrane potential exceeds a threshold level, a spike is elicited and the membrane potential is reset to its resting value.

Integrate-and-fire neuron provides a conceptually and computationally handy model, which is easy both to comprehend and simulate. This simplicity, however, can be regarded as an oversimplification for understanding the brain. How well
the integrate-and-fire neuron can model a real neuron, is a matter of debates. For instance, real neurons can generate a burst of action potentials, during which the membrane potential never decreases to the resting value. In fact, synapses are also not uni-type, but rather operate with release of various neurotransmitters. Neurotransmitters are enzymes released at synapses, which activate or deactivate the ion flow into neurons. Elemental neurotransmitters that play a role in cortex include glutamate, aspartate, gamma-aminobutyric acid (GABA), dopamine and serotonin. GABA is known to be main inhibitory neurotransmitter in the mammalian central nervous systems. Apart from neurotransmitters, existence of different ion channels and receptors as well as the 3D structure of neurons are main reasons why integrate-and-fire neuron could be a oversimplified model. In order to activate or deactivate the ion inflow, neurotransmitters bind to receptors at the receiver neurons, i.e. postsynaptic neurons. Common receptors include, for instance, NMDA and AMPA, both of which bind with glutamate.

1.2 Cortical Activity and Its Variability

Function and information of the cortex are believed to be encoded in the spatio-temporal network dynamics [Abeles 1993, Lindsey 1997, Mainen 1995, Prut 1998, Riehle 1997, Villa 1999]. Experiments with cortical recordings, however, have shown that one-to-one mapping of spatiotemporal activity patterns to behavioral and perceptual parameters is not easy. Its difficulty is mostly a result of trial-to-trial variability and nonstationarity in cortical activity [Schiller 1976, Vogels 1989, Rosenblith 1962, Arieli 1995, Arieli 1996, Tsodyks 1999]. Despite variability and non-stationary, some invariants in cortical activity patterns could be identified. One excellent example of cortical variability and underlying invariant is the work of Arieli et al. [Arieli 1996], who showed that the evoked activity in the monkey cortex extremely varied in response to a fixed stimulation across trials. The greatest amount of variability, however, could be attributed to the ongoing activity in the cortex, a process apparently distinct from the response to the stimulation. Due to its relatively higher autocorrelation, post-stimulus ongoing activity could be approximated by its value at the stimulus onset. Subtracted by the ongoing activity, the evoked activity was shown to be reproducible across trials. In other words, a simple addition of the ongoing activity to the reproducible stimulated responses yielded the evoked response (figure 1.2). The variability of the evoked response was shown to be due the variability in the ongoing activity. Note that cortical activity in that work was recorded using imaging with voltage sensitive dyes, which yielded aggregated membrane potentials of neurons in the recorded area.
Figure 1.2: Predicting the observed activity in the cortex by the invariant reproducible response and the ongoing activity. Each entry in the figure is a time dependent vector reflecting the cortical activity, which is visually recorded using voltage sensitive dyes. Figure is adapted from [Arieli 1996].

1.3 Necessity for Adaptive Methods in Modeling Neural Activity

Computational modeling approach to neuroscience has broadened our understanding of the brain by investigating the mechanisms of collective neural and synaptic activity at the network level [Dayan 2001]. Simulations of large biological neuronal networks with various architectures have helped to link structural and biophysical properties to dynamics and function [Morrison 2005, Markram 2006]. Theory and simulations of networks bring a lot of insight to how the brain works from a bottom up approach, i.e. they tackle the question, how a specific (might also be statistically described) network architecture generates the activity patterns observed in the brain, or activity patterns that is essential for its functioning. The top-down approach, on the other hand, deals with neural decoding [Dayan 2001], i.e. tackles the question what mental, behavioral and perceptual correlates of the observed neural activity are. With this approach, in vivo cortical recordings, are related to mental, behavioral or perceptual states either manually or by employing adaptive computational methods. Here, the underlying assumption is that the cortical activity reflects functional states, although it is highly variable.

Despite extensive cortical variability, cortical function remains robust to a great extent. No matter how noisy and variable the cortex is, a monkey almost always can grab a banana on the table. Obviously, there is an invariant underlying the variable cortical activity. In the work of Arieli et al., it turned out that there was a trivial relation between the perceived stimulus and the evoked activity. However, it presumably took good scientists a lot of effort of to discover that relation. For other behavioral and cognitive correlates of cortical activity, the link between the activity and the function might not be that trivial or, even trivial, it might be time consuming to discover this relation. If the stimulus is not a single type, but natural visual stimulus, manually relating the perception to cortical activity, can be too difficult. Automated adaptive computational methods can come into play to discover the invariants of to predict the functionally relevant neural activity or
activity related functional parameters. Massive amount of complex data in cortical recordings pose another big challenge to neuroscience community, which necessitates automated computer methods to make use of the data, too.

Note that in this thesis, we regard machine learning algorithms and adaptive filtering methods as adaptive computing methods. Neural decoding field has already incorporated the use of several adaptive computing methods. For instance, Mitchel et al. proposed gaussian naive bayes [Koller 2009] and support vector machine [Schölkopf 2002, Cristianini 2000a] (SVM) based methods to relate the functional Magnetic Resonance Imaging (fMRI) signals to the semantic categories of the words that subjects were reading [Mitchell 2004]. These baseline methods were improved by Hidden Process Models [Mitchell 2005, Hutchinson 2009], which assume that the observed fMRI activity is generated by multiple overlapping underlying processes. The inverse problem, i.e. predicting the fMRI brain activity, from the semantics could be also achieved [Mitchell 2008]. Unlike fMRI and imaging with voltages sensitive dyes, some recording methods yield spike data. Some of the adaptive computational methods in the literature has been especially developed for spike data. These include Expectation-Maximization like maximum likelihood methods [Chornoboy 1988, Okatan 2005b], gradient descent methods [Brown 2001] and point processing extension of online Bayesian methods [Eden 2004], which could be seen as point process extensions of Kalman filters [Maybeck 1979].

Probably, the most prominent examples of adaptive computing models for neural activity come from the area of Brain Machine Interfaces (BMI). This technology relies on a learning algorithm or an adaptive filter, expert knowledge of neural encoding about the areas and the frequency bands involved, as well as the adaptivity of the human side to the machine [Nicolelis 2001]. For instance, Mehring et al. could infer the 2D hand trajectories of rhesus monkeys from their cortical Local Field Potential (LFP) [Mehring 2003]. They employed Linear Discriminant Analysis (LDA) and SVMs for the inference. Employing SVMs with kernels that are specially designed for neural activity has also taken place in literature [Shpigelman 2002, Eichhorn 2004], with applications to brain machine interfaces [Shpigelman 2004]. Some examples also included also real-time applications [Müller 2008]. Note that SVM applications in brain machine interfaces not only utilized spike data, but also Magnetoencephalography (MEG) [Lal 2005b] and ElectroCorticography (ECoG) data [Lal 2005a]. Recently, Li et al. employed also unscented Kalman filters for BMIs [Li 2009].

As seen in the relevant literature, there is a recently increasing interest and utilization of adaptive computing methods in various subfields of neuroscience.

1.4 The "Model" Cortex

Because of difficulties of experimenting with in vivo cortices and of the attractiveness of studying anatomy-free cortices, many researchers have investigated dissociated cortical cell cultures as a model of the in vivo cortex (see chapter 2). Cortical
cultures allow for investigating the neuronal networks with large accessibility in terms of recordings and stimulations via Multi-Electrode Arrays (MEA). Since they are closed system in vitro networks, thus they do not have interference with the ongoing activity from other parts of brain. Inherent properties of cortical networks, without anatomical constraints, can be explored by utilizing cortical cell cultures and MEA setups. Cortical cell cultures are also our model networks throughout this thesis. More detailed background information about cortical cultures is presented in chapter 2.

1.5 The Modeling Tool

Machine learning algorithms build up a large branch of adaptive computing methods. Within the connectionist discipline of machine learning, recurrent neural networks constitute expressive yet difficulty trainable machinery [Haykin 1999]. In recent years, the reservoir computing approach has brought novel insight to learning with recurrent neural networks. Training only a subset of network connectivity parameters, i.e. weights, proved to be well performing yet computationally efficient in estimating time varying signals [Jaeger 2007b]. A reservoir computing architecture, namely an Echo State Network, has been shown to predict also chaotic time series [Jaeger 2004]. In order to develop adaptive computing models of cortical activity, we take a reservoir computing approach and employ Echo State Network (ESN) framework. Traditional ESN learning algorithms are not well suited for some of the subproblems tackled in this thesis. For these, we adapt the associated ESN learning algorithms for cortical culture activity. In the current thesis, we present a detailed part about reservoir computing and ESNs in chapter 5.

1.6 Motivation and Scope of This Thesis

In this thesis, we deal with functional modeling of biological neural networks. Functional modeling of in vivo cortex could be described as modeling the cortical function and dynamics from its input and initial state. Function and important parameters of dynamics could be merged into a single vector variable, which could be the "output" of the cortex. This hypothetical problem of modeling the function and dynamics of the cortex could be better formulated with the help of equations,

\[ y(t) = f^{out}(u(t), x(t)) \]

and

\[ \dot{x}(t) = f(u(t), x(t)) + \xi, \]

where \( y(t) \) stands for the the merged output variable reflecting cortical function and dynamics, \( u(t) \) denotes the cortical input, \( x(t) \) is the hypothetical cortical state that determines its changes to a great extent apart from noise \( \xi \), \( f \) and \( f^{out} \) could be any function. Note that this problem definition is so far purely hypothetical, i.e.
we do not claim that cortex can be modeled by such equations. We just utilize it for explaining functional modeling. Functional modeling of cortex would be defined as identifying functions \( f \) and \( f^{out} \). In the broad sense, neural decoding can be seen as functional modeling, provided that the model for decoding can deal with dynamic trajectories and it has a dynamical memory. (see chapter 5).

Due to limitations and practical difficulties of dealing with the \textit{in vivo} cortex and in order to investigate the anatomy free properties of biological neural networks (see section 1.4 and chapter 2), we tackled the problem of functional modeling in dissociated cortical cultures. Obviously, such networks do not natural functions. Thus, we cannot relate the activity dynamics to a physiological function in random \textit{in vitro} Biological Neural Networks (BNN). Nevertheless, by studying their activity dynamics, we gain experience and information about generic network properties forming the basis of \textit{in vivo} networks. One possible approach is to assign pseudo-functions to random BNNs by artificially mapping network states to a predefined set of actions [Chao 2008] and to study functional modeling under this setting.

Throughout this thesis, we study functional modeling of cortical cultures under two different problem settings.

1. In the first setting, we regard the response spike train of a BNN to a stream of various stimuli as a very detailed characterization of its pseudo-function and aim at modeling stimulus-response relations (see chapter 6). We record the output responses of simulated and cultured BNNs to random multivariate streams of stimuli. We tackle the question \textit{whether it is possible to train an artificial neural network that predicts the response of a reference biological neural network under the applied stimulus range}. In other words, we aim at generating an equivalent network of a BNN in terms of stimulus-response relations. Given the same stimulus, the equivalent network should predict the output of the biological neural network (figure 1.3).

2. Secondly, we associate a cultures’ function with the trajectory of its aggregated spiking rate. The spike rates over all the active MEA electrodes are summed and make up the global firing rate of the electrode. Global firing rate in dissociated cultures is oscillatory. More specifically, it undergoes temporary
episodes of intensive spiking. These episodes are known as network bursts (see chapter 2). It has been shown that global firing rate trajectories during network bursts can be grouped into several clusters [Wagenaar 2006a]. Each of which can be seen as a separate type of a network burst. Burst types are also called dynamic attractor, as the culture activity is drawn to the the temporal patterns associated with one of the possible burst types [Wagenaar 2006a].

Our second functional modeling task includes modeling these persistent temporal patterns, i.e. attractor states, of cultures [Amit 1992, Rabinovich 2001, Leonardo 2005, Bibitchkov 2002, Romani 2006]. Here, we presume that attractor states have important functional relevance in biological neural networks. Indeed, attractor states have been claimed to be relevant to motor and cognitive functions [Stopfer 2003, Miller 1996, Barak 2007]. The task is basically to simulate the global firing rate trajectory of the culture based on a cue signal. The cue signal is an indicator for the dynamic state of the neuronal culture and contains a snapshot of spiking rates of all the recording electrodes for one time step at the burst onset. This problem definition is depicted in figure 1.4 and explained more detailedly in chapter 7. Our task specification poses a test for artificial networks’ capacity to model dynamic state trajectory in the culture. As the information is primarily encoded in the spatio-temporal network dynamics, we believe that a neural modeling tool should be able to deal with dynamic trajectories.

As seen in our problem definition, we do not aim at revealing the anatomical connectivity from recordings. We rather provide tools for functional modeling. By functional modeling, we imply a behavioral mimicking of the culture. Behavioral mimicking is defined separately for each subproblem we tackled. In task 1 (chapter 6, the behavioral mimicking is defined based on spike timings, whereas in task 2 (chapter 7) it was based on aggregated firing rates.

A functional model for biological neural networks can be useful for relating the physiological and physical determinants of its activity and thereby, can be a tool for analyzing information coding in these networks. It can also be helpful for interacting with BNNs by means of electrical stimulation. The tools we provide in this thesis can be applied in various areas of neural encoding and decoding, including brain machine interfaces and other neurotechnological applications, such as prediction and interception of epileptic events.

One approach to functional modeling problem would be to embed human expert knowledge into the model, i.e. to manually identify the functionally relevant features in the input or cue signals and to design a model based on these features. Parameters of the model could be fit to the data with adaptive algorithms. Another approach is to employ a generic network model, which can be assumed to be universal for problems of neural encoding. Here, the desired situation is that functionally relevant features are already encoded in the dynamics of the generic network model and can be easily read out by linear or log-linear models. This approach has the advantage that knowing the relevant features is not a prerequisite for modeling. In the current
1.6. Motivation and Scope of This Thesis

Figure 1.4: A problem setting for functional modeling of cortical cultures. The light blue shaded region highlights the interval, in which the model simulates the burst trajectory. The solid blue line is the actual burst trajectory, whereas the red dashed line is the model output. Note that the simulation is based on the cue signal. The cue signal is given to the model at one time step, just before the light blue shaded region.

thesis we take this approach. We employ Echo State Networks (ESN) as a reservoir computing tool and propose ESN based learning tools for functional modeling of biological neural networks. Reservoir computing tools can be sensitive to various relevant features such as precisely timed input spikes, spike rates and correlations in the input or cue signal.

We list the factors that favor the usage of reservoir computing methods as follows:

**Universality** In reservoir computing, Liquid State Machines (LSM) were proposed as universal computers for tasks that require fading memory [Maass 2002]. Although the physical realization of such an LSM is not practical, the perfect theoretical LSM has the universal computing power (see chapter 5).

**Computational power** The reservoir computing tools have been shown to perform well in identifying nonlinear systems and predicting chaotic systems [Jaeger 2003, Jaeger 2004] (see chapter 5).

**Biological plausibility** Biological neural networks, in vivo, process a continuous stream of inputs in real time. Moreover, they prove successful to react and compute fast, independent of their instantaneous state and ongoing activity, when prompted by sudden changes in stimuli. In other words, they perform any time computing [Maass 2004]. Reservoir computing has also been suggested as a model of cortical information processing for their capability of
online and anytime computing, for their fading memory and for their separation properties. It has been argued that specific cortical circuitry is possible to build into generic LSM framework [Maass 2004]. Bringing reservoir computing into the problem, thus, does not only deliver expressive models that can distinguish a rich set of input patterns, but also provides more biological relevance to the theoretical tool.

**Online learning** Both reservoir computing tools, ESNs and LSMs can be applied in an online learning setting. This might gain crucial importance as the dynamics of in vivo activity change with time.

**Fitness for spike data** Traditional ESN algorithms are intended for continuous valued signals. Spike data, however, is naturally better modeled under a point process framework (see chapter 3). ESN algorithms can be adapted for point processes. In task 1 of our problem specification, functional modeling includes modeling the output spikes of the neuronal culture. In chapter 6, we explain how we adapted ESN learning algorithms for point process data.

**Inherent oscillatory dynamics** In order to model the global firing rate trajectories based on a single cue signal, the modeling tool must be able to generate oscillations. Echo State Networks, as recurrent networks, have inherent oscillatory dynamics, if their feedback connections are sufficiently strong. In fact, oscillations have been shown to be an inescapable property of recurrent networks with sufficient feedback [Amit 1992].

**Being Handy** ESN algorithms usually do not require extensive preprocessing of the data as the reservoir can be sensitive to different forms of input signals under appropriate parameters. Fine-tuning the initial reservoir parameters, however, might be time consuming.

### 1.6.1 Findings and contributions

In the current thesis, as our major contribution, we propose reservoir computing based adaptive tools for functional modeling of cortical cell culture activity. Our findings mediated via this thesis can be summarized as follows:

- We show that it is possible to deliver predictive functional models of dissociated cortical cultures (chapters 6 and 7).
- We adapt the ESN readout learning algorithms for point process data based on point process log-likelihood (chapter 6).
- We bring our proposed ESN learning algorithms beyond readout learning and show how to efficiently adapt the reservoir parameters based on the propagation of the point process log-likelihood into the reservoir. Note that in a recurrent network, back-propogation based algorithms are computationally expensive. For a sparse network and point process data, however, we show that gradients can be well approximated with one-step propagation (chapter 6).
We propose feed-forward reservoirs and demonstrate that sparse feed-forward reservoirs are better adaptable by one-step log-likelihood propagation than recurrent ones. We also analyze the underlying reasons for that and experimentally show that gradients are better approximated by one-step log-likelihood propagation in feed-forward reservoirs (chapter 6).

We show that models for dynamic attractor states of dissociated cortical cultures can be developed by reservoir computing methods. The global firing trajectory could be related to a cue signal (chapter 7).

Although our main focus is functional modeling of biological neural networks, we also propose an adaptive computing method for mining the structure of biological neural networks. We propose the NeuronRank algorithm, for extracting structural features from a biological neural network. NeuronRank assigns source and sink values to every neuron in the network in an iterative scheme. The proposed algorithm is neural analogous of the link mining algorithms PageRank [Page 1998] and Hubs&Authorities [Kleinberg 1999]. Using NeuronRank, we show that adaptive computing methods are not restricted to neural activity but they can also extend to neural connectivity (chapter 8). Using network simulations, we show that NeuronRank extract features that are predictive for activity parameters of the network.

1.6.2 How is the rest of this thesis organized?

In the rest of this thesis, we proceed as follows: In chapter 2, we provide a review of the relevant literature about dissociated cortical cultures. This chapter provides the background information about the biological aspect of this thesis, which would later be helpful for biological interpretation of our results. Chapter 3 gives a brief introduction to point process modeling of neural systems and builds the foundations for chapters 4 and 6. In chapter 3, we propose a naive point process based online prediction method for spike activity in neuronal cultures. The method proposed essentially applies online stochastic gradient descent to spike recordings in neuronal cultures. Chapter 6 presents a reservoir computing approach to point process modeling of cultures. In contrast to naive point process modeling, the reservoir computing approach provides a stationary functional model of neuronal culture activity. In chapter 7, we present our ESN-based functional modeling methodology for simulating firing rate trajectories in network bursts. Note that this part of the work is not only about mapping an input stream to an output stream online. The input is presented as a cue at one time step. The ESN then simulates the burst trajectory by utilizing a feedback of its output to its input. Chapter 8 presents a link-mining approach to the structure of biological neural networks. The work in that chapter also presents an adaptive computing approach to biological neural networks. The focus, however, is on relating the structure to activity rather then activity to function. Finally, in chapter 9 we interpret the results from different parts of the thesis together.
Chapter 2

Cortical Cell Cultures

Contents

2.1 Introduction ........................................... 13
2.2 Culture Preparation ................................. 15
2.3 Electrophysiology ........................................ 15
2.4 Activity and Dynamics ............................... 16
2.5 Pharmacological Treatments of Neuronal Cultures ....... 19
2.6 Induced Changes and Learning in Neuronal Cultures ............ 19
2.7 Artificial Functions for Neuronal Cultures .................. 20

2.1 Introduction

Human behavior, or more generally behavior of mammals, is a product of a very sophisticated brain architecture. It is commonly thought that mammalian brains achieve their high degree of learning, adaptability, variability and complex behavior particularly due to their cerebral cortex, as this part is found only in mammals. *In vivo* experiments have revealed the relation between cortical activity and cognitive states, perceptual states as well as motor behavior. *In vivo* recordings, however, are associated with experimental drawbacks due to the small ratio of recorded neurons to the whole local network, difficulties of recording a sufficient number of single units and interference with ongoing activity from other parts of the brain. Studying learning, memory and other behavior related phenomena can be hampered by these drawbacks, which have motivated the research on cortical cell cultures. Mammalian cortical cells can be disassociated and regrown outside an organism, such that they develop into a neuronal cell culture. They build closed system *in vitro* living networks that are used to study inherent physiological properties of neural systems [Marom 2002].

Although the anatomical structure of the brain is not preserved in dissociated cultures, as networks of real neurons, they have desirable properties that favor the study of information processing, learning and memory with them. Marom and Shahaf have summarized these properties in their review article on cortical cultures [Marom 2002]. Cultures form extensive connectivity and various types of activity patterns in their maturation process. It has been argued that these properties are essential for learning and memory. Unlike *in vivo* networks and acute cortical
Figure 2.1: A photo image of a neuronal culture with MEA electrodes. We aim at generating an equivalent network in terms of input-output relations. Photo was taken by Steffen Kandler from BCCN Freiburg.
slices, dissociated cultures do not have preexisting anatomical constraints. This allows for investigating the capabilities and limits of networks made by biological networks independently from a specific architecture and a broader view linkage between connectivity and activity. Isolation from external processes and other parts of the brain is also an important motivation for working with cell cultures. Interference with ongoing brain activity can be avoided in cultures so that the system properties and response characteristics can be better studied. Moreover, maturation process of a dissociated culture displays high structural plasticity. Neuronal cultures allows one to study how a biological neural network forms starting from isolated neurons. Compared to in vitro brain slices, cultured networks are more accessible in terms of ratio of the recorded neurons to the whole network. In other words, they pose a less serious under-sampling problem. It has been proposed and believed that discovering principles of activity dynamics, adaptation, and signal processing in cultures is important, as the experience and knowledge gained here might be transferred to that of real brains.

2.2 Culture Preparation

Cells are usually harvested from prefrontal cortices of neonatal rats before the brain fully matures. A cortical culture developing on a dish might have up to 5000 neurons/mm$^2$ [Marom 2002]. Synapses do not exist directly after dissociation but start to develop within hours. A mature culture, which is around 4 weeks old in vitro, holds extensive number of active synapses. Despite the dissimilarity of the connectivity to the in vivo case, the distribution of the cell types are similar to that of in vivo cortex.

2.3 Electrophysiology

Electrophysiological techniques including patch clamping, extracellular recordings of multi-unit activities as well as calcium imaging can be employed in the study of cortical cultures. A particularly strong and well-suited tool for dissociated cultures is Multi-Electrode Arrays (MEA) [Marom 2002]. MEAs have recently been employed to both stimulate and record from neuronal cultures. They allow for simultaneous recordings and stimulations from up to 60 electrodes from a culture of around 300 mm$^2$ surface area. Each electrode picks up the extracellular electrical field of one or several neurons. Since neurons transmit information by action potentials, or spikes, one needs additional preprocessing to extract these spikes from the extracellular signals via adaptive high-pass filtering and thresholding. Such tools have been developed and are widely used [Egert 2002]. Considering the spatial resolution of stimulations and recordings, MEAs provide valuable insights into information processing in neuronal cultures and brain slices.
Chapter 2. Cortical Cell Cultures

2.4 Activity and Dynamics

Dissociated cultures display spontaneous activity. This activity starts as action potentials that are asynchronous across electrodes around one week after preparation [Marom 2002, Kamioka 1996]. Through the maturation process, during which cultures build synapses, activity becomes synchronous across electrodes. More specifically, activity is composed of irregular network-wide bursts of spikes. Cultures display little or no spiking activity most of the time (that is, between bursts) and extremely high spiking activity during the bursts. Figure 2.2 shows a sample activity pattern of a neuronal culture.

Although the inter-burst intervals are not necessarily regular, there seem to be spatio-temporal patterns that occur within bursts [Feber 2007, Rolston 2007]. For example, a burst might always start with the activity of the same channel and continue with the activity of another particular channel. Figure 2.3 shows a plot of structured activity within a burst.

Even though it is reasonable to think that network bursts start due to spontaneous fluctuations in the network, their propagation is directed by the underlying network connectivity. Maeda et al. studied the spatial characteristics and propagation velocities of network bursts in developing cultures [Maeda 1995]. They reported that frequency and the propagation velocities of network bursts increased significantly with the maturation process, with frequencies increasing from 0.01 to 0.5 Hz and velocities increasing from 5 to 100 mm/sec. Bursts had randomly various onset locations, supporting a spontaneous fluctuation hypothesis as their driving force. Maeda et al. have managed to physically divide the culture into several parts using an ultraviolet laser. As a consequence, they observed bursts with different frequencies in each part. Furthermore, the burst had different phases in each part.

Figure 2.2: Spike activity in neuronal cultures. The activity is composed of irregularly occurring network bursts. Each dot indicates a spike detected on the corresponding electrode (y-axis) at the corresponding time (x-axis).
2.4. Activity and Dynamics

Figure 2.3: Spike activity in neuronal cultures. Zoom into a burst. Each dot indicates a spike detected on the corresponding electrode (y-axis) at the corresponding time (x-axis)

i.e. their onset times were not locked to each other. A small physical opening between two compartments resulted in delayed propagation of the burst from the original source compartment into the other compartment. These findings indicate that bursts in the culture propagate as a consequence of network connectivity. Connectivity synchronizes the random fluctuations in different parts of the culture into synchronized network bursts.

Jimbo et al. also studied the characteristics of activity in dissociated cortical cultures, with a focus on responses to electrical stimulation [Jimbo 2000]. They reported that the culture responds to a simple voltage stimulation with two phases, being the early and late phases. The early phase is composed of a precise spatio-temporal pattern that hardly varies across trials, whereas the late phase display a high trial-to-trial variability. Adding NMDA blockers to the cultures suppressed and even eliminated the late phase, whereas they did not show any effect to the early phase. Jimbo et al. argue that early and late phases are governed by distinct synaptic pathways and the dynamics of the culture is multi-modal, including deterministic and stochastic/chaotic modes.

Wagenaar et al. reported the richness of the bursting dynamics in dissociated cultures [Wagenaar 2006b]. Upon their analysis of around 60 cultures, they found that the bursting patterns were extremely various within each culture in terms of rate profiles, temporal structure, spatial organization of activity. They claim that the richness of the structure in burst patterns is suggestive for the important role of burst in information processing during development. There is also evidence that suppression of bursting prevents mature network formation, indicating that bursts are needed for structural maturation of cortical networks.

Although there is a rich repertoire of bursts in cortical cultures, these bursts can be grouped into clusters, each cluster displaying its own spatio-temporal pat-
terns of activity. Rolston et al. [Rolston 2007] showed that there is "a robust set of spontaneously repeating spatiotemporal patterns of neural activity" in cortical cultures using a template matching algorithm. This observed "robust set" of activity patterns remains stable over minutes of recording. Rolston et al. argue that the neural network spontaneously self-organizes itself into these patterns, which are intrinsic properties of any self-organizing network.

Beggs and Plenz describe the activity in neuronal cultures from the perspective of criticality and information transmission [Beggs 2003]. They observed and reported that the activity propagation in dissociated cortical cultures can be described by the same equations that describe avalanches. Based on the similarity of the mathematical descriptions, they regard synchronous bursts as neuronal avalanches. Similar to avalanches and earthquakes, in neuronal avalanches the sizes of the activity obey power laws. Characteristics of this power law show that the culture is in a self-organized critical state, where information transmission is optimized. According to Beggs and Plenz, this self-organized criticality can be a generic property of cortical networks and can be called a new mode of activity beside oscillatory, synchronized and wave-like modes.

Wagenaar et al. [Wagenaar 2006a] regarded the burst activity in neuronal cultures from a dynamical systems perspective and reported that their activity is self-organized into dynamic attractors. They investigated the evolution of aggregated firing rates over all electrodes during network bursts. By employing a two-variable analysis of the firing rate, i.e. rate at a time point versus rate at 500 ms before that time point, they demonstrated that phase plot of the culture activity falls into one of the several distinct trajectories, each of which being a dynamic attractor. Driven by its spontaneous fluctuations, culture activity is drawn to one of these attractors. A combined trajectory of multiple attractors was argued to be resulting from super-burst, which is a successive frequent occurrence of different bursts. Note that each of these dynamic attractors correspond to a separate type of network burst.

Stability and breakableness of network bursts have also been a focus of interest in the cultured cortical networks area. Wagenaar et al. [Wagenaar 2005] have stopped bursting in neuronal cultures using multi-electrode stimulation. They hypothesized that the reason for bursting is the lack of external input to neuronal cultures, unlike the case in vivo, where a local neural circuit gets afferent inputs from other brain areas. In accordance with other related work, they report that slow stimulation via single electrode elevates the burstiness of the culture. With stimulation via several distributed electrodes cyclically, however, the bursts could be controlled, i.e. burstiness could be significantly reduced. A closed loop system, where the strength of stimulation was fine-tuned according to culture activity was a part of the experimental setup. The strength was arranged to make a constant tonic firing rate in the culture. Wagenaar et al. argue that one of the reasons that cultures burst is that they are closed loop systems and they lack natural input.

Cultures have shown to carry significant information about the type of external stimuli in their temporal order of the events. Shahaf et al. have recently shown that the order, in which the electrodes record their first action potentials in re-
response to an external stimulus, provides sufficient information about the stimulus location [Shahaf 2008]. This is an example of order based representation in neuronal cultures, which is also existent in vivo [Shahaf 2008, Lee 2002]. The findings in this work indicates that the information is not only in rates or in action potential times, but also in the spatial onset orders in the culture. This fact has the interesting impact that the same information about the external variants can be represented in different time scales in the cultures, more specifically independent of the time scale of the activity involved in the representation.

2.5 Pharmacological Treatments of Neuronal Cultures

Pharmacological agents have played also a considerable role in neuronal culture research. One example is tetrodotoxin, which eliminates the electrical activity in the culture. In order to study how electrical activity shapes the structure of a network, controlled experiments with tetrodotoxin were performed [Marom 2002, van Huizen 1985, Huizen 1987]. It was shown that cultures manipulated with tetrodotoxin, having abolished electrical activity, displayed enhanced neural outgrowth and branching. Control (pharmacologically not manipulated) cultures displayed extensive eradication of synapses at the beginning of their mature phases. Tetrodotoxin added cultures, on the other hand, did not have this fingerprint of maturation onset. Even the removal of tetrodotoxin did not result in synapse elimination, indicating a 'critical period', after which the effect of the pharmacological treatment is irreversible.

2.6 Induced Changes and Learning in Neuronal Cultures

One further motivation to study neuronal cultures is understanding their properties, which are traditionally attributed to animals and computers, e.g. learning and memory. Several experiments have been performed to investigate how the external stimuli can induce structural changes in the cell cultures, which could be the correlate of the memory for the presented stimuli or the correlate of the learning to react to a certain type of stimulus. One common finding is that repetitive high amplitude single electrode stimulation increases the responsiveness of the culture to a weaker stimulus [Jimbo 1998, Maeda 1998]. An inspiring example is Jimbo et al. [Jimbo 1999], who reported that a high frequency stimulation at 20 Hz, i.e. tetanic stimulation, enhanced or depressed the response to selected test stimuli. Moreover, this potentiation or depression was not neuron specific but pathway specific. In other words, for a selected tetanic stimulus, either all of the recorded neurons increased or all of them decreased their response to test stimuli. The same tetanic stimulus could both enhance and depress the response. Cross-correlation analysis revealed that correlation of the ongoing activity with tetanus induced activity decided on whether potentiation or depression would take place. If the ongoing activity before the tetanus was correlated to tetanus induced activity, potentiation of the pathway was observed.
Contrarily, response depression was observed as a consequence of uncorrelated ongoing activity with tetanus induced activity. A probable underlying mechanism for the finding is that a pathway is strengthened or depressed depending on whether it is active at the time of tetanic stimulation. That the potentiation/depression has the same type for all the neurons, is possibly due to the highly recurrent architecture, which has homogenous influence on individual neurons.

A further example of learning or adaptivity has been demonstrated by Shahaf and Marom [Shahaf 2001]. They proposed a stimulation procedure for a neuronal culture, in which a local low frequency stimulation is repeated until a predefined criterion for the response was fulfilled. Whenever the desired response was detected, the procedure stopped itself. Shahaf and Marom found out that for further epochs of the procedure, the number of repetitions needed to fulfill the criterion was smaller. The learning curve, i.e. number of repetitions for each epoch, revealed that the culture was reinforced to deliver the desired response more quickly upon training. The reinforcing drive for the culture was the sudden absence of the stimulus, rather than its onset. A theory in behavioral psychology states that removal of a drive might act as a reward to an organism [Hull 1943, Shahaf 2001]. The reported phenomenon by Shahaf and Marom shows that this theory might also hold for neuronal cultures, without the need for a body.

Ruaro et al. [Ruaro 2005] provided another example of neuronal culture training, which could be related to image processing. They trained a neuronal culture to respond distinguishably to two different input patterns. Patterns were an 'L' and an inverted 'L', which were spatially presented to the neuronal culture. The 'L' input was the simultaneous stimulation of the electrode at the left and the bottom edges of a MEA, whereas the inverted 'L' input was that of right and top edge electrodes. Tetanic stimulation was delivered to the culture in order to potentiate a response to the selected pattern. Upon training, the action potential rates in response to these two different patterns were significantly different. Their work gives hints about how intrinsic adaptivity of a biological neural network might help designing neuronal computers with extensive and natural associative capabilities.

2.7 Artificial Functions for Neuronal Cultures

A neuronal culture is dissociated from the brain tissue. It is regrown on a dish and hence does not have a function, i.e. its activity does not have cognitive, motor and behavioral correlates. This is a major drawback for investigating the functional significance of their structural and physiological properties. This prompted several studies, where artificial functions are assigned to neuronal cell cultures. One eye-catching example of assigning functionality to a neuronal culture is due to De Marse et al. [DeMarse 2001]. De Marse et al. introduced the "neurally-controlled animat", an agent in a virtual environment, whose motor commands are generated by the signals of a MEA neuronal culture. Sensory feedback is presented from virtual environment to the neuronal culture in terms of electrical stimulation. Here, the
computer acts as a translator between the animat and the culture, translating sensory feedback into appropriate stimulations and neural activity into culture to one of the possible distinct motor commands. The long term goal of the animat project is to embed learning into the system. Another work in this area is due to Bakkum et al., who extended the "neurally-controlled" animat to reinforcement learning setting, where a predefined patterned stimulus train was presented to the culture as a reward to desired response \cite{Bakkum2008}. Conversely, a shuffled, unpatterned frozen noise stimulus was presented as a consequence of not desired activity.
3.1 Introduction

Action potentials in a neuron are generated in an all-or-none fashion. In other words, spikes elicited by any particular neuron have a uniform shape. This uniformity of spikes allow for a simpler modeling of neuronal outputs than modeling of a continuous time signal. Ignoring the fact that spikes have a time span, i.e. there is a time difference between their beginning and end, the simpler mathematical framework for modeling spikes is that of temporal point processes. A temporal point process is a stochastic signal of binary events that occur in continuous time \[ \text{Daley 2003} \]. Hence, signals that are comprised of point-like events in time, e.g. trains of action potentials generated by neurons, can be characterized in terms of stochastic point processes. Examples of point processes include Poisson processes, where event occurrences are independent of the event history; and renewal processes, where inter-event intervals are mutually independent. More generally, occurrence of a point event can depend on the full history of the point process. This case would necessitate a mathematical description that relates the point event occurrences to event history in terms of a conditional intensity function, which is defined as

\[
\lambda(t|H_t) = \lim_{\Delta \to 0} \frac{P(N(t + \Delta) - N(t) = 1|H_t)}{\Delta},
\]
where \( N(t) \) is the cumulative event count. \( \lambda(t|H_t) \) represents the conditional probability density that a spiking event occurs at time point \( t \). \( H_t \) stand for the event history of the process. The conditional intensity function fully characterizes a point process [Daley 2003]. The conditional intensity function can be also described in terms of inter-event-time probability density,

\[
\lambda(t|H_t) = \frac{P(p(t|H_t))}{1 - \int_0^t p(u|H_u) \, du},
\]

where \( p(t|H_t) \) denotes the inter-event probability density [Brown 2003a]. Probability of having at least one spike in any time interval \([t, t + \Delta)\) can be expressed in terms of conditional intensity function,

\[
Pr(\text{spike in } [t, t + \Delta) | H_t) = \lambda(t|H_t) \Delta + o(\Delta),
\]

where \( o(\Delta) \) accounts for occurrence of more than one event in the interval. If \( \Delta \) is sufficiently small so that it can contain no more than one event,

\[
Pr(\text{a spike in } [t, t + \Delta) | H_t) \approx \lambda(t|H_t) \Delta.
\]

The above equality is called the Bernoulli approximation to point-processes.

### 3.2 Point Process Log-Likelihood

The value of the joint probability density function for a sample counting path in interval \([0, T)\) is expressed as [Brown 2003a],

\[
p(N_{0:T}) = \prod_{j=1}^J \lambda(u_j|H_{u_j}) \exp\{-\int_0^T \lambda(u|H_u) \, du\}
\]

\[
= \exp\{\int_0^T \log \lambda(u|H_u) \, dN(u) - \int_0^T \lambda(u|H_u) \, du\}. \tag{3.1}
\]

Here, \( u_1, ..., u_j, ..., u_J \) are the observed event times in interval \([0, T)\). Let \( \theta \) be the set of parameters in the conditional intensity function, \( \lambda \). Then, log-likelihood for the sample counting path can be trivially derived from equation 3.2,

\[
\mathcal{L}(\theta|N_{0:T}) = \log p(N_{0:T}|\theta)
\]

\[
= \int_0^T \log \lambda(u|H_t, \theta) \, dN(u) - \int_0^T \lambda(u|H_u, \theta) \, du.
\]

Note that this expression can be reformulated as

\[
\mathcal{L}_i(\theta|N_{0:T}) = \int_0^T \ell(u, \theta) \, du,
\]
one obtains

$$\ell(u, \theta) = \log \lambda(u|H_u, \theta) \frac{dN(u)}{du} - \lambda(u|H_u, \theta),$$  \hspace{1cm} (3.3)

the instantaneous log-likelihood for time point $u$ [Brown 2003a].

### 3.3 Generalized Linear Models and Point Process Modeling of Neural Systems

In neural systems, the activity history is not the only factor that affects action potential generation. The activities of other neurons, physiological parameters and external stimulation can all affect the output of a neuron. Although the conditional intensity function was proposed to link a point process to its history, it can be extended in a straightforward way to account for all other variables that act on the point process. Let $\lambda_i$ be the conditional intensity for the output spiking process of neuron $i$. Then the log-likelihood for counting path $N_i$ for neuron $i$ is expressed as [Chornoboy 1988]:

$$\mathcal{L}_i(\theta|N_i) = \log p(N_i|\theta) = \int_0^T \log \lambda_i(t|I_t, H_t, \theta) dN_i(t) - \int_0^T \lambda_i(t|I_t, H_t, \theta) dt, \hspace{1cm} (3.4)$$

where $p(N_i|\theta)$ is the conditional probability density for the counting path and $I_t$ stands for the set of all other variables that act on neuron $i$.

The quality of point process modeling strongly depends on the expressive power of function $\lambda(t|I_t, H_t, \theta)$. Linear or log-linear models might be preferable for better parameter fitting, whereas nonlinear functions allow for more expressive models. Point process models of neural activity have mostly incorporated the spike history and external variables through generalized linear models, i.e. the models that link the independent variables to probability distributions using a linear operator and a nonlinear link function [Paninski 2004, McCullagh 1989]. If $X$ denotes the vector containing all independent variables,

$$p(\text{spike}_i(t)|X, \theta) \sim \lambda_i(t|X, \theta) = f_\theta(AX), \hspace{1cm} (3.5)$$

where $A$ is a linear operator that maps the vector of independent variables to a lower dimensional space and $f_\theta$ is the link function. A very common selection for $f_\theta$ is the exponential function [Paninski 2004].

### 3.4 Adaptive Point Process Modeling

There have been several approaches for learning the parameters of the conditional intensity function, $\lambda$. For instance, Brown et al. [Brown 2001] have proposed adapting the parameters of the conditional intensity function by gradient descent learning.
26 Chapter 3. Point Process Modeling of Neural Systems

### 3.4.1 Gradient Descent Learning

With a conversion to discrete time, i.e. $t \rightarrow n$, $\Delta \rightarrow 1$, equation 3.3 for instantaneous log-likelihood becomes,

$$\ell_i^n = \log \lambda_i^n \delta_{N_i} - \lambda_i^n,$$  

where $n$ is the discrete time step index. Any parameter $\theta_j$ of the model is then learned by the standard gradient descent learning.

$$\theta_j^{n+1} \leftarrow \theta_j^n + \eta \frac{\partial \ell_i^n}{\partial \theta_j},$$

where $\eta$ is the learning rate. For the sake of readability, one can leave the calculation points of partial derivatives out and simply use the notation $\theta_j^{n+1} \leftarrow \theta_j^n + \eta \frac{\partial \ell_i^n}{\partial \theta_j}$. Using equation 3.6, the partial derivative becomes

$$\frac{\partial \ell_i^n}{\partial \theta_j} = \frac{1}{\lambda_i^n} \frac{\partial \lambda_i^n}{\partial \theta_j} \delta_{N_i} - \frac{\partial \lambda_i^n}{\partial \theta_j}.$$  

Gradient descent learning rule has been employed for deriving models of receptive fields in rat hippocampal neurons [Brown 2001]. Note that this learning rule inherently allows for online adaptation of model parameters.

### 3.4.2 Maximum Likelihood Approaches

Gradient descent algorithm can be also applied in batch mode rather than online mode,

$$\theta_j^{n+1} \leftarrow \theta_j^n + \eta \frac{\partial \mathcal{L}_i^n}{\partial \theta_j},$$

Gradient descent algorithm will converge to the maximum likelihood estimate of the parameter vector only under certain conditions. Log-likelihood must be a convex function with respect to the parameters in order algorithm to converge to its true global maximum. Paninski has showed that log-likelihood in equation 3.4 converges to its global maximum with gradient descent if $f_\theta(B)$ of equation 3.5 is jointly convex in $\theta$ and $B$ and $\log f_\theta(B)$ is jointly concave in $\theta$ and $B$. This condition implies the log-concavity of the likelihood [Paninski 2004].

Another maximum likelihood approach is due to Chornoboy et al. and Okatan et al. [Chornoboy 1988, Okatan 2005b]. Their approach regard parameter learning in point process models as a statistical inference problem rather than a gradient descent one. Chornoboy et al. proposed an Expectation-Maximization [Dempster 1977] based algorithm, that iteratively increased the likelihood of the model.

### 3.4.3 Adaptive Point Process Filtering

Online Kalman filtering [Maybeck 1979] approaches have also been adopted for point process data. For instance, Eden at al. online introduced point process adaptive
3.4. Adaptive Point Process Modeling

filtering [Eden 2004]. The major contribution of adaptive point process filtering is online adaptation of the model parameters based on Bayesian inference. This online Bayesian approach is particularly well suited for cases where the observed system is nonstationary, i.e. certain parameters change with time. The underlying assumption is that the parameters of the system parameters evolve with a linear operator and Gaussian noise

\[ \theta_{k+1} = F_k \theta_k + \eta_k. \]

Here, \( F_k \) is the system evolution matrix and \( \eta_k \) is a zero-mean white noise with covariance \( Q_k \). Let \( \Delta N_k \) be the number of spikes that are observed in the interval \((t_{k-1}, t_k]\). If \( \Delta t_k = t_k - t_{k-1} \) is small enough so that \( \Delta N_k \) is either 1 or 0, the probability of observing \( \Delta N_k \) spikes can be well approximated by the Bernoulli approach,

\[ \Pr(\Delta N_k \text{ spikes in } (t_{k-1}, t_k] | \theta_k, H_k) = \exp(\Delta N_k \log(\lambda(t_k|\theta_k, H_k)\Delta t_k) - \lambda(t_k|\theta_k, H_k)), \]

where \( H_k \) is a concatenated vector of the spike and parameter (\( \Delta N \) and \( \theta \)) history. Applying Bayes’ law to get the posterior distribution on the parameters

\[ p(\theta_k | \Delta N_k, H_k) = \frac{p(\Delta N_k | \theta_k, H_k)p(\theta_k | H_k)}{p(\Delta N_k | H_k)}, \quad (3.7) \]

and deriving the prior distribution over the parameters from the previous posterior distributions using the Chapman-Kolmogorov equation,

\[ p(\theta_k | H_k) = \int p(\theta_k | \theta_{k-1}, H_k)p(\theta_{k-1} | \Delta N_{k-1}, H_{k-1})d\theta_{k-1}, \]

yields a computationally tractable expression on the posterior parameter distribution. Here, \( p(\theta_k | \theta_{k-1}, H_k) \) is determined by the system evolution matrix \( F_k \) and noise \( \eta_k \). Combining the predicted parameter distribution from the system evolution equation and the observed \( \Delta N_k \) with Bayesian inference 3.7 yields the following online prediction scheme on parameter vector \( \theta \). Here, prior and posterior distributions of \( \theta \) are assumed to be multivariate Gaussian. \( W_{k|k} \) denotes the covariance estimate of the parameter vector upon observing \((t_{k-1}, t_k]\). All the estimates subscripted with \( k|k \) denote the estimates upon observing the \( k \)-th interval, whereas those subscripted \( k|k-1 \) denote the estimates predicted by the system evolution equation

\[ \theta_{k|k-1} = F_k \theta_{k|k-1} \]

\[ W_{k|k-1} = F_k W_{k-1|k-1} F_k' + Q_k \]

\[ (W_{k|k})^{-1} = (W_{k-1|k-1})^{-1} + \left[ \frac{\partial \log \lambda}{\partial \theta_k} \right]' \left[ \lambda \Delta t_k \right] \left( \frac{\partial \log \lambda}{\partial \theta_k} \right) - (\Delta N_k - \lambda \Delta t_k) \frac{\partial^2 \log \lambda}{\partial \theta_k \partial \theta_k'} \theta_{k|k-1} \]
\[ \theta_{k|k} = \theta_{k|k-1} + W_{k|k} \left[ \frac{\partial \log \lambda}{\partial \theta_k} \right]' (\Delta N_k - \lambda \Delta t_k) \theta_{k|k-1} \]

One of our main contributions in the current work is adapting reservoir learning algorithms for point process data. We present the details of this contribution in chapter 6.
# Chapter 4

## Spike Prediction in Neuronal Cultures

This chapter investigates the question whether it is possible to predict the upcoming spikes in a neuronal culture. The recorded spike train at an electrode is modeled as a point process, where the firing probability depends on the finite spike history of all electrodes. To capture the potential plasticity of the network, we employ a gradient descent method, which naturally allows for online learning. Several experiments with different cultures show that learned models can predict upcoming spike activity quite well. With this approach, we also construct a functional connectivity map between the electrodes.

## 4.1 Introduction

Mammalian cortical cells can be dissociated and regrown outside an organism, such that they develop into a neuronal cell culture. They are closed system \textit{in vitro} living networks, which are frequently used to study physiological properties of neural systems [Marom 2002] (see chapter 2). Although the anatomical structure of the brain is not preserved in cultures, their inherent properties as networks of real neurons, their accessibility and small size make them very attractive for investigating information processing in biological networks.

Recordings of spontaneous activity from neuronal cultures have shown that the activity is composed of irregular network-wide bursts of spikes [Marom 2002]. These networks display little or no spiking activity most of the time (that is, between

<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Introduction</td>
</tr>
<tr>
<td>4.2 The Model</td>
</tr>
<tr>
<td>4.2.1 Modeling the spike activity as point processes</td>
</tr>
<tr>
<td>4.3 The Learning Algorithm</td>
</tr>
<tr>
<td>4.4 Experimental Results</td>
</tr>
<tr>
<td>4.4.1 Evaluation of the model with binary prediction</td>
</tr>
<tr>
<td>4.4.2 Implications of the results</td>
</tr>
<tr>
<td>4.5 Conclusion</td>
</tr>
</tbody>
</table>
bursts) and extremely high spiking activity during the bursts. Figure 4.1 shows a sample activity pattern of a neuronal culture.

The question addressed in this chapter is whether it is possible to derive abstract, interpretable and biologically meaningful predictors of spatio-temporal activity patterns in a neuronal cell culture. This question is interesting, as a positive answer might contribute to a deeper understanding of the mechanisms of activity propagation, and its results might help controlling the activity in living neural networks. Here, we propose an online algorithm that learns a predictor based on a functional connectivity model of spontaneous neuronal activity in tissue cultures, which is basically the effective map of time dependent influences between the recorded units.

Although it is reasonable to think of 'spontaneous' fluctuations which start a burst in these cultures, the spatio-temporal spread of activity is nevertheless generated and shaped by the underlying network. It is then an interesting problem to predict the upcoming activity in these models, as the predictor model can reflect the underlying connectivity. Obtaining an activity predictor for biological neuronal systems, including neuronal cultures, is challenging in many ways. Their activity dynamics display both nonlinear and non-stationary characteristics. Noise is another innate property of these systems causing high variability of the activity. The combination of these properties suggests the use of automatized adaptive methods, i.e. machine learning algorithms, to obtain such models.

Spike activity can be conceptualized as event data, i.e. time series of point events occurring in continuous time such as the arrival of buses to a bus station or failure times of light bulbs. A further example of such data is alerts in telecommunication networks, which might occur in various types at any time [Mannila 1997]. Mannila et al. proposed an algorithm that can find frequently occurring episodes (sequences of events). Apart from mining frequently occurring sequences, point processes provide
Figure 4.2: Spike activity in neuronal cultures. Zoom into a burst. Each dot shows a spike detected on the corresponding electrode (y-axis) at the corresponding time (x-axis)

4.1. Introduction

A mathematical framework for modeling such data. A point process [Cox 1980] is a stochastic process of event times \((t_1, t_2, ..., t_N)\), where \(N\) is the number of events and \(t_i\) are continuous variables. There have been various research efforts to model neural spiking activity as point processes. For further work and background information about point process modeling, we refer chapter 3.

The focus of this chapter is an online algorithm that learns a predictor based on a functional connectivity model of spontaneous neuronal culture activity. The learned model could also be interpreted as an abstraction of the underlying effective network connectivity, i.e. its functional connectivity. The functional connectivity model is similar to the one in Chornoboy et al. [Chornoboy 1988] and in Okatan et al. [Okatan 2005a]. We take a gradient descent approach for learning the functional connectivity, which is able to capture network plasticity, e.g. changes in its structure. This is different from the maximum-likelihood approach [Chornoboy 1988] [Okatan 2005b], as we aim at online learning that is able to catch up with a changing structure. We use the instantaneous log-likelihood of a point process as a criterion to improve the model. This approach has previously been suggested to analyse neural receptive field plasticity [Brown 2001]. Here we apply it to simultaneous parallel multi-channel recordings from neuronal cultures, demonstrate its use for predicting upcoming spike activity. We provide a detailed experimental section that shows that such an approach performs well in binary prediction as well as in conditional intensity estimation using a ROC curve analysis for several different networks.

As depicted in figure 4.3, we recorded the self-sustained activity in neuronal cultures with Multi-Electrode-Arrays (MEA) and extracted the spike data using the MEA Toolbox [Egert 2002]. We then fed the spike data into our learning algorithm. The algorithm produced a predictive computational model of cell culture activity. We assessed the quality of the model by testing its predictive performance.
4.2 The Model

With the current technology it is not possible to record the activity of the whole network at cellular resolution. Instead, Multi-Electrode-Arrays can record local field potentials from 60 different sites sitting on a regular grid. We can estimate the network structure of the neuronal culture to the level that the resolution of the recording allows. We do not aim at reconstructing neuron-to-neuron connectivity, but rather at learning a map of effective influences that the recorded units (single neurons or local groups of neurons) exert on each other, that is we are estimating the functional connectivity.

Here, we regard Functional Connectivity as an abstract description of the underlying structure and employ it as a predictive model of neuronal culture activity to the degree it can be observed.

We model the functional connectivity by a network of $C$ nodes, where $C$ is the number of electrodes that record activity from the culture. For each ordered pair of nodes $(i, j)$ and each admissible delay $d$, the model has a parameter $w_{ijd}$ which represents the effect of a spike in node $j$ on the input of node $i$ after time $d$. Figure 4.4 depicts this model of functional connectivity graphically.

The net input to node $i$ is then the linear sum of all inputs with all possible delays

$$o_i(t) = \sum_j \sum_d w_{ijd} u_j(t - d)$$

(4.1)

where $o_i(t)$ is the net input to unit $i$ at time $t$, $d$ is running through all possible delays, and $u_j(t - d)$ is the output of unit $j$ at time $t - d$ (1 if there is a spike, and
4.2. The Model

Figure 4.4: Modeling functional connectivity

The net input of a node is related to its spiking activity through a sigmoidal function

\[ \lambda_i(t|H_t) = f(o_i(t)) = \frac{1}{\exp(-o_i(t)/A) + 1} \] (4.2)

where \( \lambda_i(t|H_t) \) is the instantaneous rate of spiking of unit \( i \), \( H_t \) is the spike history up to time \( t \) and \( f \) is a sigmoidal function with parameter \( A \).

4.2.1 Modeling the spike activity as point processes

Spikes are usually conceived as point-like events in time. A spiking process is hence modeled as a point process, which in turn is fully characterized by its conditional intensity function [Cox 1980].

The firing rate is defined as a conditional intensity function

\[ \lambda_i(t|H_t) = \lim_{\Delta \to 0} \frac{P(N_i(t + \Delta) - N_i(t) = 1|H_t)}{\Delta} \]

where \( N_i(t) \) is the cumulative spike count of unit \( i \). \( \lambda_i(t|H_t) \) represents the conditional probability that a spiking event occurs in an infinitesimal time interval. Note that if \( \Delta \) is small enough such that no more than one spike event can be observed during it, then

\[ P(N_i(t + \Delta) - N_i(t) = 1|H_t) = \lambda_i(t|H_t) \Delta \]

The log-likelihood of a sample count path for a node (electrode) \( i \) is expressed as [Chornoboy 1988]

\[ \mathcal{L}^i(\theta|N) = \log P(N_i|\theta) = \int_0^T \log \lambda(t|H_t) \, dN_i(t) - \int_0^T \lambda(t|H_t) \, dt. \]
The instantaneous log-likelihood \([\text{Brown 2001}]\) of a point process trajectory (spike train) is defined as

\[
\ell_{it}(\theta) = \log \lambda_i(t|H_t) \frac{dN_i(t)}{dt} - \lambda_i(t|H_t)
\]  

(4.3)

Note that

\[
\mathcal{L}_i(\theta|N) = \int_0^T \ell_{it} dt.
\]

4.3 The Learning Algorithm

Gradient descent approach and online learning

We take a gradient descent approach to optimize the parameters. For each learning step, we modify the parameters such that we follow the steepest descent in the negative log-likelihood landscape

\[
w_{ijd}^{t+1} = w_{ijd}^t + \eta \frac{\partial \ell_{it}}{\partial w_{ijd}},
\]

(4.4)

where \(\eta\) is a constant learning rate.

In this chapter, we propose an algorithm that has a batch learning phase, and an online learning phase. We employ the online learning phase in order to catch up with the drifting parameters of the activity dynamics.

In tables 4.1 and 4.2, a sketch of the algorithm is given. We split our spike data into two distinct spike trains, namely the training and test spike trains. The training spike train is reserved for batch learning, whereas the test spike train is reserved for both online learning and evaluation of the predictive performance. The batch learning phase runs repeatedly from the beginning to the end of the training spike train, whereas online learning phase goes through the test spike train only once. Note that two spike trains have nothing to do with each other. In the test spike train, parameter update at a time point is performed only after the conditional intensity for that time point has been estimated. This guarantees that we do not use the information from this time point for learning before prediction.

4.4 Experimental Results

In our experiments, we evaluated the predictive performance of the learned functional connectivity model for the upcoming spike activity. Since we define functional connectivity as a map of effective influences, we assess the quality of the model according to its predictive power.

In order to test the quality of the functional connectivity estimation, we evaluated our intensity estimation for the last 200 seconds of recordings with respect to observed spiking activity of the cultures. We discretized the time space with 1 ms bins, the size also used for discretizing the delay space, i.e. the discretization step
4.4. Experimental Results

Table 4.1: Batch learning

repeat
  increase iteration index \( k \)
  for each output unit \( i \), input unit \( j \) and delay \( d \)
    \( \Delta w_{ijd}^k = 0 \)
  endfor
endwhile

\( time = INITIAL\_TRAINING\_TIME \)
while \((time < END\_OF\_TRAINING\_TIME)\)
  for each output electrode \( i \)
    compute \( \lambda_i(time|H_{time}) \) (equation 4.2)
    for each input unit \( j \) and delay \( d \)
      \( \Delta w_{ijd}^k \leftarrow \Delta w_{ijd}^k + \frac{\partial \ell_i}{\partial w_{ijd}} W_k \)
    endfor
  endfor
  \( time \leftarrow time + 1 \)
endwhile
\( w_{ijd}^k \leftarrow w_{ijd}^{k-1} + \eta \Delta w_{ijd}^k \)
until insignificant increase in the likelihood

Table 4.2: Online Learning

\( time = INITIAL\_TEST\_TIME \)
while \((time < END\_OF\_TEST\_TIME)\)
  for each output electrode \( i \)
    compute \( \lambda_i(time|H_{time}) \) (equation 4.2)
    for each input unit \( j \) and delay \( d \)
      \( w_{ijd}^t \leftarrow w_{ijd}^{t-1} + \eta \frac{\partial \ell_i^t}{\partial w_{ijd}} W_{t-1} \)
    endfor
  endfor
endwhile
was 1 ms in equation 6.2. For each upcoming time bin we estimated the conditional intensity of the spiking process from the spike history, as in equation 4.2. Based on estimates for all bins (during network bursts) in those last 200 seconds, we performed a Receiver Operating Characteristic (ROC) curve analysis and computed the area under the ROC curve. ROC curves are extensively used in signal detection theory and machine learning. In a binary classification problem, a ROC curve is a plot of true positive rate vs. false positive rate. In contrast to classification accuracy, the area under a ROC curve (AUC) is invariant to prior class probabilities [Bradley 1997]. It should be noted that activity in neuronal cultures is typically a sequence of network bursts. Only very few spikes occur outside the bursts. Therefore, the algorithm performs learning and prediction only during bursts. If there are at least 100 ms between two succeeding groups of spikes, we regard them as different bursts. If a spike is neither followed nor preceded by another spike within 30 ms (before it and after it), it is regarded as an isolated spike (outside of any burst) and dismissed from the analysis. If an electrode has never spiked in the training spike train (for batch learning), it is regarded as an inactive electrode and dismissed from the learning process.

We experimented with five different neuronal cultures and computed the areas under ROC curves for all the active electrodes in those cultures. All the cultures included inactive electrodes. The areas under ROC curves for a selected neuronal culture are shown in figure 4.5. Note that these electrodes are the active ones that have spiked at least 10 times within the evaluation duration of 200 seconds. The areas under ROC curves were generally close to 1, indicating a good estimation performance.

Figure 4.5: culture 133: Areas under ROC curves for 1 ms prediction windows. 0 entries are displayed for inactive electrodes.
4.4. Experimental Results

Figure 4.6: culture 133: precision, recall, accuracy and AUC (first half of the electrodes), for a 30ms prediction window

Figure 4.7: culture 133: precision, recall, accuracy and AUC (second half of the electrodes), for a 30ms prediction window
Table 4.3: Accumulated results over all active electrodes. The column AUC shows the average area under ROC curves for a prediction window of 1 ms. Also ratio of positives, precision, recall and f-measure are shown for a prediction window of width 30 ms with a comparison to perceptron results.

<table>
<thead>
<tr>
<th>window</th>
<th>30 ms</th>
<th>Point Process Based</th>
<th>Perceptron</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 ms</td>
<td>30 ms</td>
<td>30 ms</td>
</tr>
<tr>
<td>culture</td>
<td>pos AUC prec. rec. f-m.  prec. rec. f-m.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>0.16</td>
<td>0.78 0.53 0.54 0.53</td>
<td>0.46 0.45 0.45</td>
</tr>
<tr>
<td>133</td>
<td>0.16</td>
<td>0.88 0.67 0.67 0.67</td>
<td>0.63 0.63 0.63</td>
</tr>
<tr>
<td>135</td>
<td>0.16</td>
<td>0.75 0.63 0.64 0.63</td>
<td>0.17 0.53 0.22</td>
</tr>
<tr>
<td>138</td>
<td>0.28</td>
<td>0.78 0.73 0.73 0.73</td>
<td>0.63 0.59 0.61</td>
</tr>
<tr>
<td>146</td>
<td>0.16</td>
<td>0.63 0.62 0.63 0.63</td>
<td>0.37 0.30 0.33</td>
</tr>
</tbody>
</table>

4.4.1 Evaluation of the model with binary prediction

Although the learning algorithm estimates the conditional intensity only, i.e. an instantaneous rate, one can extend the problem to the case where a binary target value is predicted. In particular, one may employ the algorithm to predict whether a channel will spike within a predefined time interval. In such a case one can evaluate the prediction accuracy, ‘precision’ #True Positives/(#True Positives + #False Positives) and ‘recall’ #True Positives/(#True Positives + #False Negatives), as well as ROC curves.

We state the binary prediction problem as follows: At a time point $t$, given the spike history $H_t$ and the size $W$ of the prediction window, predict whether there will be at least one spike in interval $(t, t + W]$. To adapt the algorithm for this problem, we convert the original spike train $S$ to $S'$. For each event of time point $t_i$ in $S$, $S'$ includes events of the same type at time points $\{t_i - W, t_i - W + \Delta, t_i - W + 2\Delta, \ldots, t_i\}$, where $W$ is the prediction window size and $\Delta$ is the discretization time. Note that in this problem, the algorithm uses spike history $H_t$ from $S$, whereas the target $\frac{dN}{dt}$ is the increase in the count of points of process $S'$. Obviously, if $\frac{dN}{dt} \geq 1$, then there is at least one spike in interval $(t, t + W]$ for unit $i$. The results of binary prediction for a particular culture are shown in figures 4.6 and 4.7. Note that for all figures, the channels not shown were the silent ones, i.e. those which displayed no or very little spike activity during the recordings. We maintain that our results indicate meaningful functional connectivity models.

Additionally, we provide aggregated results for all electrodes, for different neuronal cultures. These results are shown in table 4.3. The table shows the averaged areas under ROC curves for 1 ms prediction windows, and also, average precision, recall and f-measure values for a 30 ms prediction window. The ratio of the positive examples are also presented in the column 'Pos.'. The rightmost part of the table is shown for comparison, where we implemented a simple perceptron threshold unit based on the same functional connectivity model (i.e. for each pair of nodes and each
4.4. Experimental Results

4.4.2 Implications of the results

Implications of our results are manifold. Firstly high areas under ROC curves show that cell culture activity is not random and exhibits a rather tightly defined spatio-temporal structure. At the same time, areas under ROC curves indicate that valid functional connectivity models can be learned via gradient descent online learning in a point process framework. Note that the functional connectivity model depends on certain parameters. Estimation of the conditional intensity for a prediction window of 1 ms and for a binary prediction window of 30 ms yields different models. This is acceptable, since functional connectivity is the map of effective influences, the nature of which depends on the circumstances. As the time scale of the influence changes, one must expect that the learned model also changes.

As an example, we present the functional connectivity parameters for prediction widths of 1 ms, 5 ms, 15 ms and 30 ms between two selected units in figure 4.8. Although delay dependences of the parameters display some cross similarity between different prediction window widths, fine structures of these dependences and parameter values differ. Figure 4.8 also depicts an example of the fact that the excitatory/inhibitory character of the functional connectivity depends on its associated delays. The parameters for the same input-output pair might have different signs at different delays.

We also evaluated the dependence of the prediction performance on the prediction window width. Figure 4.9 shows aggregated results of this evaluation for all the active electrodes in a selected culture. Not surprisingly, the ratio of the positive time windows (i.e. those containing at least one spike) increases with the window width. Precision and recall also improve with the window width as they are sensi-
Figure 4.9: Prediction performance depending on the width of the prediction window. Shown are the aggregated results over all the active electrodes in culture 133.

tive to the ratio of positives. On the other hand, areas under ROC curves (AUC), which are invariant to prior class probabilities, remain stable over different window widths. Because of the stable AUC, we believe that the increase in precision and recall is due to more balanced prior class probabilities rather than directly due to the time scale of the prediction.

4.5 Conclusion

We have addressed the question of whether it is possible to derive abstract and interpretable predictors of spatio-temporal activity patterns in a neuronal cell culture and we have also provided a positive answer to this question. We proposed a functional connectivity model and an online algorithm to learn it from measured spiking activity in neuronal cultures. The learned model was also employed as a predictor for the upcoming spike activity. We presented experimental results that show that the algorithm generally learns to predict spiking activity quite well.
5.1 Introduction

Recurrent neural networks (RNN), i.e. neural networks with feedback loops, have been used as associative memories and input-output mapping systems [Haykin 1999]. Using dynamical systems terminology, associative memories act by converging to a fixed point attractor, when driven by a static input. Biologically motivated learning rules, e.g. Hebbian learning, have motivated successful examples of associative memories such as Hopfield Network [Haykin 1999, Hopfield 1982]. Input-output mapping systems act by responding a time-varying input signal and ideally map them to a desired output signal. Such systems are also named as dynamical memories [Lukosevicius 2009]. Input-output mapping RNNs function as nonlinear filters and can be analyzed using dynamical systems framework. RNNs are very expressive in modeling dynamical systems. In other words, they can cover a very large set of models by employing different parameters and topologies. In fact, they were shown to universally approximate dynamical systems under certain assumptions, i.e. they can simulate a large subset of dynamical systems [Maass 2007, Funahashi 1993]. Furthermore, they
are universal computers in the sense that they can simulate any *Turing machine* [Haykin 1999, Siegelmann 1995].

In spite of their expressive power, RNNs have been considered as problematic tools for machine learning and identifying dynamical systems due to their adversities in parameter learning. Although significant work has been done on recurrent learning, such as *Back-propagation Through Time* [Werbos 1990] and *Real Time Recurrent Learning* [Williams 1989], these algorithms had practical drawbacks. These algorithms are gradient descent based algorithms and extensions of back-propagation algorithm, which is traditionally used to train feedforward neural networks. Gradient computations can be computationally very expensive due to cyclic topology. Besides, RNNs, as dynamical systems, might go through bifurcations during recurrent learning [Doya 1992]. Bifurcations would in turn result in the discontinuity in the system characteristics, which are hard to deal with gradient based learning algorithms. Note that a learning dynamical memory should deal with long term dependencies, i.e. it has to capture the effect of the input signal for long horizons in the future. Such long term dependencies were shown to be difficultly captured by gradient descent [Bengio 1994].

A fundamentally different approach for using RNNs as dynamical memories is *reservoir computing*. More recently, Liquid State Machines (LSM) and Echo State Networks (ESN) have been introduced as efficiently learning recurrent neural networks. [Jaeger 2001, Maass 2002]. The common key contribution of these approaches is the proposal of a recurrent neural network with a fixed connectivity, i.e. a *reservoir*, which does not have stable states and has a fading memory of the previous inputs and network states. In response to an input stream, the reservoir generates a higher-dimensional spatio-temporal dynamics reflecting the structure in the input stream. The higher dimensional reservoir state can be mapped to a target output stream online, with a second module, namely a readout. As each reservoir neuron becomes sensitive to different features of the input stream, the readout module maps these distinguished features to the output stream. As the readout module is composed of a single layer topology, learning only at the readout layer does not suffer from the drawbacks of recurrent learning. LSMs are networks of spiking *integrate-and-fire* (IAF) neurons whereas ESNs use continuous valued sigmoid neurons and a single layer of readout neurons (see 7.2).

### 5.2 Echo State Networks

Echo State Networks (ESN) [Jaeger 2001] consist of a recurrent neural network (RNN) reservoir and a readout module (figure 7.2). Because recurrent neural networks (RNN) are in general difficult to train with gradient descent methods [Bengio 1994], the *echo state* approach to RNNs is motivated by the observation that a fixed (untrained) RNN can be very useful for discriminating multivariate time series if the RNN is gradually forgetting internal states and thereby also the inputs at
5.2. Echo State Networks

previous time steps. This property is described as generating the echo states of the input. Existence of echo states is independent of the specific structure of the reservoir, but strongly determined by the extension of the eigenvalues of its connectivity matrix. The input stream is used to stimulate a higher dimensional dynamical system, which can be mapped to a target output stream by adaptive linear or sigmoidal readout units. Even with random fixed reservoirs, ESNs make up a successful reservoir computing framework with applications in several engineering and nonlinear modeling tasks [Jaeger 2003, Jaeger 2006, Jaeger 2004]. They have been successfully applied for models of per periodic attractors [Jaeger 2006], energy saving in wireless communications [Jaeger 2004], nonlinear system identification [Jaeger 2003] and functional modeling of biological neural networks [Gürel 2009].

5.2.1 Description of Echo States

Let \( x^n \) denote the vector whose entries denote the activation states for individual units a RNN at time step \( n \). Similarly, let \( u^n \) denote the vector of dynamic drive to this network, i.e. its input stream at time step \( n \). For any causal system, \( x^n \) is a function of \( u^n, u^{n-1}, u^{n-2}...u^{n-h} \), where \( h \geq 0 \). If \( x^n \) can be uniquely determined by \( u^{n-h} \), i.e. it is not influenced by \( u^{n-h} \) for large \( h \), then the activation state vector \( x^n \) can be intuitively regarded as an echo of its input history [Jaeger 2001].

5.2.2 Dynamics

The ESN dynamics is described as

\[
\begin{align*}
x^{n+1} &= f(W^{\text{in}} u^{n+1} + W x^n + W^{\text{back}} y^n) \\
y^{n+1} &= f^{\text{out}}(W^{\text{out}} [u^{n+1}; x^{n+1}; y^n]),
\end{align*}
\]

where,
$W : N \times N$ internal reservoir weight matrix,
$W^{in} : N \times K$ input weight matrix,
$W^{out} : L \times (K + N + L)$ output weight matrix,
$W^{back} : N \times L$ output weight matrix,
$x^n : N \times 1$ state vector for time step $n$,
$u^n : K \times 1$ input vector for time step $n$,
$y^n : L \times 1$ output vector for time step $n$,
'$:$ vertical vector concatenation.

\[ f = (f_1, ..., f_N)^T \] are nonlinear saturating functions, where \( f_i(x) = \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \). \( f^{out} = (f^{out}_1, ..., f^{out}_L)^T \) are the output functions for each output unit.

5.2.3 Echo State Property

Echo state property defines the existence of echo states for a neural network [Jaeger 2001]. A network has echo states if there exists input echo functions \( E = (e_1, ..., e_N) \), where \( e_i : U^{-N} \rightarrow \mathbb{R} \) such that for all left-infinite input histories ..., \( u^{n-1}, u^n \in U^{-N} \) the current network state is

\[ x^n = E(..., u^{n-1}, u^n). \]

Here \( U^{-N} \) is the space of all possible left-infinite input sequences. The definition states that left-infinite sequences are the unique determinants for the network state. The same definition implies that whatever the current state of the network is, a right-infinite input sequence will determine network state asymptotically. Moreover, any pair of states will converge to the same state under infinite inputs. A network holding these properties are called state contracting and state forgetting, respectively [Jaeger 2001]. Both properties are equivalent to having echo states. Also as a consequence of the definition, if two input sequences are close enough with respect to an Euclidian distance on \( \mathbb{R}^N \), then the resulting echo states are also close with respect to the same distance (continuity property). We refer to [Jaeger 2001] for the formal proposition and the proof. This property is also called fading memory in another reservoir computing framework, namely Liquid State Machines [Maass 2002].

5.2.4 Existence of Echo States

Sufficient conditions for the existence/non-existence of echo states could be given based on the Lipschitz property of the weight matrix [Jaeger 2001].

Let the network units have sigmoidal functions \( f = \tanh \), the largest singular value of weight matrix \( W \) be \( \sigma_{max} = \wedge \) and the eigenvalue of weight matrix \( W \) with the largest absolute value be \( \lambda_{max} \). Assuming that the network has no feedback connection from readout to the reservoir,

\begin{itemize}
  \item [a)] if \( \wedge < 1 \), then \( d(T(x, u), T(x', u)) < \wedge d(x, x') \) for all inputs \( u \) for all states \( x, x' \in [-1, 1]^N \). In this case echo states exist for all inputs \( u \) for all states \( x, x' \in [-1, 1]^N \),
\end{itemize}
b) if $\|\lambda_{\text{max}}\| > 1$, then the network has an asymptotically unstable null state. In this case it has no echo states for any input set $U$ containing 0.

We refer to [Jaeger 2001] for the formal proposition and the proof.

### 5.2.5 Training Echo State Networks

Training ESNs is in most cases restricted to learning $W^{\text{out}}$, i.e. the weight from the reservoir to readout units. When the readout units are linear, the training can be performed in batch mode upon harvesting the reservoir states driven by the input stream. Learning typically succeeds by linear regression, e.g. using the pseudo inverse matrix

$$W^{\text{out}} = Y [X; U]^+,$$

where $L \times T$ matrix $Y$ is the collection of target vectors $y$ for $T$ time steps, $X$, $U$ are the same collection for reservoir states and input vectors. $^+$ is pseudo inverse operation.

Much more efficient is the use of the Wiener-Hopf equation in terms of computational space and time complexity [Jaeger 2007a]. Let $[X; U] = X$,

$$W^{\text{out}} = p R_x^{-1},$$

where $R_x^{-1} = XX^T$ and $p = YX^T$.

Apart from error minimization, Steil [Steil 2004] proposed the backpropagation-decorrelation algorithm, which seeks a trade-off between error minimization and decorrelation of the reservoir activity.

#### 5.2.5.1 Learning in the Reservoir

The most important merit of reservoir learning is the sufficiency of the readout training for many applications. Though this did not hinder several attempts to combine this property with supervised and unsupervised reservoir learning algorithms in order to significantly increase the prediction or regression quality [Lukosevicius 2009]. Unsupervised learning methods preadapt the reservoir with respect to some goodness measure other than its prediction performance. These include eigenvalue spread [Farhang-Boroujeny 1998], pairwise correlations and entropy of the activity distribution [Jaeger 2005]. There are also other strategies for unsupervised reservoir adaptation resulting from biological motivations. Preadapting the reservoir with Hebbian and Anti-Hebbian learning are the ones with no success [Jaeger 2005]. Steil, on the other hand, reported that adopting the intrinsic plasticity rule from real biological systems improved the performance of the backpropagation-decorrelation algorithm [Steil 2007]. In the scope of this thesis, we propose a reservoir adaptation method for point process outputs using log-likelihood of the output process as a goodness measure and approximating the log-likelihood gradient [Gürel 2009]. We refer to chapter 6 for the details of this method.
5.2.6 Echo State Networks with Leaky Integrators Neurons

One extension of ESNs comes from upgrading the nature of the individual neurons. In the models described until now, the neurons are memoryless, i.e. their states of activation depend only on their current inputs. This limitation (or simplicity) relieved by introducing leakage time constants to neurons, i.e. making them leaky integrators. Leaky integrator neurons act as low pass filters. We refer these networks as Leaky Integrator Echo State Networks (LI-ESN). LI-ESN dynamics is described with

\[
x^{n+1} = (1 - \frac{1}{\tau}) x^n + \frac{1}{\tau} f(W^{in} u^{n+1} + W x^n + W^{back} y^n), \quad \tau \geq 1.
\]

\[
y^n = f^{out}(W^{out} [u^n; x^n]),
\]

where \(\tau\) is a global parameter for all reservoir neurons. Note that these equations are discrete time conversions of the dynamics of a network whose units exponentially discharge under 0 input using Euler discretization [Jaeger 2007a].

LI-ESNs have been shown to work well on noisy systems with slowly changing dynamics and are especially useful for time warped data [Jaeger 2007a]. The leakage time constant is another aspect of fading memory. For instance, even a purely feed-forward reservoir will have a fading memory due the memory of the individual neurons.

5.2.7 Feed-Forward Echo State Networks

Introducing time constants into reservoir units eliminates the necessity of a recurrent network for dynamic memory. Even there is no feedback connection in a network of leaky integrator neurons, an impulse will produce dynamic network activity for a while, depending on to leakage time constants. Note that as long as \(\tau \geq 1\), this memory is necessarily fading and the network will have echo states. Therefore, acyclic random neural networks with leaky integrator neurons can be used as reservoirs [Gürel 2009]. We refer an ESN with such a reservoir as a Feed-Forward Echo State Network (FF-ESN).

Recurrent ESNs are intuitively more expressive than FF-ESNs. In other words, it can be expected that a recurrent ESN with the same number of neurons discriminates between more input patterns than a FF-ESN. In the case of adaptive reservoirs, however, we showed that a point-process-likelihood-based reservoir adaptation algorithm works especially better in feed-forward reservoirs than in recurrent ones [Gürel 2009]. Having an algorithm that is more fit to a feed-forward topology is a good reason to use feed-forward ESNs. We refer to chapter 6 for FF-ESNs.
5.3 Liquid State Machines

While Echo State Networks represent the engineering approach to reservoir computing, Liquid State Machines (LSMs) have been proposed as a model for information processing in the brain. The term liquid state machine refers to the similarity of the machine states to those of a liquid. The metaphor is based on the fact that liquids have only one stable state, namely the resting state. The perturbations on the surface of a liquid, however, are transient states that reflect the past and current disturbances that are exerted on the liquid’s surface. When there are no disturbances acting on the surface anymore, the liquid will settle to the resting state. In other words, the liquid has a fading memory of the past inputs. No attractor states can be encountered on a liquid, however, its transient states are reflections of the forces exerted as liquid perturbations [Maass 2002].

5.3.1 Dynamics

The work on LSMs is motivated by the idea that even randomly connected spiking neurons have useful property of acting as a liquid reservoir of dynamic stimulus. Figure 5.2 depicts the architecture of an LSM [Maass 2002]. The dynamic stimulus is denoted by $u(\cdot)$ and represents the disturbances on the liquid surface. Here, $y(\cdot)$, a function of time, represents the target function. The machine maps $u(\cdot)$ to $y(\cdot)$. At every time step $t$, the machine generates its liquid state $x^M(t)$ from $u(s)$, where $s \leq t$. Let $L^M$ be the filter that maps input functions $u(\cdot)$ to functions $x^M(t)$. Then dynamics of a liquid state machine can be described as

$$x^M(t) = (L^M u)(t).$$

$$y(t) = f^M(x^M(t)).$$

Here, the mathematical description filter $L^M$ depends on the physical realization the reservoir and $f^M$ denotes the readout function. For a successful map of input streams to target output streams, significantly different input stream functions should result in significantly different reservoir states.

5.3.2 Computational Power of Liquid State Machines

LSMs were proposed as devices to have universal computers for tasks that requires fading memory. Maass et al. define the universal computational power in [Maass 2002] as

We say that a class of machines has universal power for computations with fading memory on functions of time if any filter $F$, i.e., any map from functions of time $u(\cdot)$ to functions of time $y(\cdot)$, that is time invariant and has fading memory, can be approximated by machines from this class, to any degree of precision.
Figure 5.2: Architecture of a LSM. A function of time (time series) $u(.)$ is input to the liquid filter $L^M$. $x^M(t)$ is the liquid state at time $t$. $f^M$ denotes memoryless readout function that maps liquid states to the target output $y(t)$. Figure adapted from [Maass 2002].

LSMs have this universal computational power if the liquid filter has Separation Property, $SP$ and the readout map has Approximation Property, $AP$ [Maass 2002]. More precisely, LSMs can approximate any time invariant filter with fading memory if their traces of internal states differ at least for one time point in response to any two different input streams (Separation Property) and if the readout modules can approximate any continuous function from $\mathbb{R}^m \rightarrow \mathbb{R}$, where $m \in \mathbb{N}$ (Approximation Property) [Maass 2002]. Note that it is practically difficult to have a physical LSM realization that has $SP$ for any kind of input streams. For many computational tasks, however, $SP$ can be pronounced enough to achieve the task. Satisfying the separation property for a particular computation depends on whether the reservoir is composed of sufficient basis filters. A random reservoir needs to have a rich repertoire of basis filters in order to approximate the target time-invariant filter. This could be achieved for several tasks with sufficiently large random reservoirs [Maass 2002].

5.3.3 Further remarks on LSMs

LSM reservoirs consist of spiking neurons, most commonly leaky integrate-and-fire neurons. LSM reservoirs have been shown to simulate a large class of higher order differential equations with appropriate feedback and readout functions [Maass 2007]. Biological neural networks, in vivo, process a continuous stream of inputs in real time. Moreover, they prove successful to react and compute fast, independent of their instantaneous state and ongoing activity, when prompted by sudden changes
in stimuli. In other words, they perform *any time computing*. LSMs were capable for *any time computing*, indicating another motivation for LSMs to be a model for cortical information processing. Although, the cortex has very specific architecture that might not fall into a random network category, it has also been argued that specific cortical circuitry is possible to build into generic LSM framework [Maass 2004]. LSMs are both strong tools for analyzing the data from biological networks and biologically plausible models for biological neural networks.
Chapter 6

Functional Modeling of Biological Neural Networks using Reservoir Adaptation for Point Processes

Contents

6.1 Introduction .................................................. 52
  6.1.1 Dissociated cortical cultures as reference biological neural networks ............................................. 53
  6.1.2 Problem Statement ........................................ 53
6.2 The Echo State Network Model .............................. 55
  6.2.1 ESN with Leaky Integrators ............................. 55
  6.2.2 Feed-Forward Echo State Networks .................... 57
  6.2.3 ESN Learning ............................................. 57
6.3 Point Process Modeling of Spike Data ...................... 58
6.4 Point Process Modeling with ESN and Log Likelihood Propagation .............................................. 59
  6.4.1 Reservoir Adaptation ...................................... 59
  6.4.2 Existence of Local Maxima and Confidence Intervals ................................................................. 61
6.5 Evaluation of the Learned Models ......................... 62
6.6 Experimental Results ......................................... 62
  6.6.1 Simulations of Random Cortical Networks .......................... 62
  6.6.2 Prediction of Spontaneous Events in Neuronal Cultures .............................................................. 73
  6.6.3 Next-Event Prediction in Neuronal Cultures ......... 74
6.7 Conclusion ..................................................... 80

The complexity of biological neural networks does not allow to directly relate their biophysical properties to the dynamics of their electrical activity. In this chapter, we present a reservoir computing approach for functionally identifying a biological neural network, i.e. for building an artificial system that is functionally equivalent to the reference biological network. Employing feed-forward and recurrent networks with fading memory, i.e. reservoirs, we propose a point process based learning algorithm to train the internal parameters of the reservoir and the connectivity between the reservoir and the memoryless readout neurons. Specifically,
the model is an Echo State Network (ESN) with leaky integrator neurons, whose individual leakage time constants are also adapted. The proposed ESN algorithm learns a predictive model of stimulus-response relations in in vitro and simulated networks, i.e. it models their response dynamics. Receiver Operating Characteristic (ROC) curve analysis indicates that these ESNs can imitate the response signal of a reference biological network. Reservoir adaptation improved the performance of an ESN over readout-only training methods in many cases. This also held for adaptive feed-forward reservoirs, which had no recurrent dynamics. We demonstrate the predictive power of these ESNs on various tasks with cultured and simulated biological neural networks.

6.1 Introduction

As a result of extensive experiments of cortical recordings, it has been widely postulated and accepted that function and information of the cortex are encoded in the spatio-temporal network dynamics [Abeles 1993, Lindsey 1997, Mainen 1995, Prut 1998, Riehle 1997, Villa 1999]. The right level of describing the dynamics, however, is a matter of intensive discussions [Prut 1998, Rieke 1999]. Are spike rates or spike timings more relevant? What is the right temporal precision if the latter proves significant? What should be the spatial resolution of this description? Is information alternatively encoded in the firing order of the neurons in time [Shahaf 2008, Thorpe 2001, Johansson 2004, Richmond 2004]? How far can the population activity of neurons be related to function or behavior? Does the correlated activity of multiple neurons indicate a functionally relevant state?

Depending on the answers to the above questions one would preferably apply different models to relate the network activity to function. Another approach is to employ a generic network model, which can be assumed to be universal for problems of neural encoding. The parameters of the model would be learned by adaptive algorithms. Obviously, such a model should be able to deal with single spikes with high temporal precision as well as population rates. It should also be able to catch, with the appropriate parameters, network synchrony and polychrony [Izhikevich 2006]. A good candidate for this approach is reservoir computing.

With the appropriate parameters, reservoir dynamics can be sensitive to different features of the input such as correlations, polychronous and synchronous spikes, different frequency bands or even temporally precise single spikes. For instance, LSMs can approximate any time invariant filter with fading memory if their traces of internal states differ at least for one time point in response to any two different input streams (Separation Property, SP) and if the readout modules can approximate any continuous function from $\mathbb{R}^m \rightarrow \mathbb{R}$, where $m \in \mathbb{N}$ (Approximation Property, AP) [Maass 2002]. Satisfying the separation property depends on whether the reservoir is composed of sufficient basis filters. A random reservoir needs to have a rich repertoire of basis filters in order to approximate the target time-invariant filter. This could be achieved for several tasks with sufficiently large random reser-
voirs [Maass 2002]. Furthermore, reservoirs have been shown to simulate a large
class of higher order differential equations with appropriate feedback and readout
functions [Maass 2007]. These findings suggest that reservoir computing can be used
as a generic tool for the problems of neural modeling. Another factor that favors
the use of reservoir computing for neural modeling problems is biological plausibil-
ity. Reservoir computing has also been suggested as a model of cortical information
processing for their capability of online and anytime computing, for their fading
memory and for their separation properties. It has been argued that specific corti-
cal circuitry is possible to build into generic LSM framework [Maass 2004]. We refer
to chapter 5 for a detailed review of reservoir computing.

6.1.1 Dissociated cortical cultures as reference biological neural
networks

While brain tissue has highly specialized architecture and developmental history,
generic biological networks can be created as cell cultures of mammalian cortical
neurons that have been dissociated and regrown outside an organism. They are
closed system in vitro living networks, which are frequently used to study physio-
logical properties of neural systems [Marom 2002]. Using cultured networks elimi-
nates the problem of interference with ongoing activity from different parts of the
brain. Compared to in vitro brain slices, cultured networks are more accessible
in terms of ratio of the recorded neurons to the whole network. In other words,
they pose a less serious under-sampling problem. Studying such networks can pro-
vide insight into the generic properties biological neural networks, independent of
a specific anatomy [Marom 2002]. The activity in neuronal cultures is composed
of irregular network-wide bursts of spikes, even in absence of an external stimula-
tion [Marom 2002](Figure 6.2). We refer to chapter 2 for detailed review of cultures.

6.1.2 Problem Statement

Although we cannot relate the activity dynamics to a physiological function in ran-
dom in vitro BNNs, by studying their activity dynamics we gain experience and
information about generic network properties forming the basis of in vivo networks.
Moreover, one can assign pseudo-functions to random BNNs by artificially mapping
network states to a predefined set of actions [Chao 2008]. One can also regard the
response spike train of a BNN to a stream of various stimuli as a very detailed char-
acterization of its pseudo-function and aim at modeling stimulus-response relations.
In the present work, we take this approach. We record the output responses of sim-
ulated and cultured BNNs to random multivariate streams of stimuli. We tackle the
question whether it is possible to train an artificial neural network that predicts the
response of a reference biological neural network under the applied stimulus range.
In other words, we aim at generating an equivalent network of a BNN in terms of
stimulus-response relations (figure 6.3). Given the same stimulus, the equivalent
network should predict the output of the biological neural network.
Chapter 6. Functional Modeling of Biological Neural Networks using Reservoir Adaptation for Point Processes

Figure 6.1: A photo image of a neuronal culture with MEA electrodes. We aim at generating an equivalent network in terms of input-output relations. Photo was taken by Steffen Kandler from BCCN Freiburg.

Figure 6.2: Spike activity in neuronal cultures. Burst activity (left). Zoom into a burst. Each dot shows a spike detected on the corresponding electrode at the corresponding time (right). Activity recorded by Steffen Kandler.
6.2 The Echo State Network Model

In this chapter, we employ an Echo State Network with leaky integrator neurons. Echo State Networks (ESN) [Jaeger 2001] consist of a recurrent neural network (RNN) reservoir and a readout module (figure 7.2). Recurrent neural networks (RNN) are in general difficult to train with gradient descent methods [Bengio 1994]. The echo state approach to RNNs is motivated by the observation that a fixed (untrained) RNN can be very useful for discriminating multivariate time series if the RNN is gradually forgetting internal states and thereby also the inputs at previous time steps. This property is described as generating the echo states of the input. Please see chapter 5 for further details.

6.2.1 ESN with Leaky Integrators

Fading memory in the reservoir is due to recurrent connectivity if the individual neurons are memoryless. Using leaky integrator neurons results in a reservoir made from a network of low pass filters, which can be referred to as Leaky Integrator Echo
State Networks (LI-ESN). LI-ESNs have been shown to work well on noisy systems with slowly changing dynamics [Jaeger 2007a]. The leakage time constant is another aspect of fading memory. For instance, even a purely feed-forward reservoir will have a fading memory due the memory of the individual neurons. We describe the LI-ESN dynamics with

\[ x^{n+1} = (1 - \frac{1}{\tau}) x^n + \frac{1}{\tau} f(W^{in} u^{n+1} + W x^n), \quad \tau \geq 1. \]

\[ y^n = f^{out}(W^{out}[u^n; x^n]), \]

where,
- \( W : N \times N \) internal reservoir weight matrix,
- \( W^{in} : N \times K \) input weight matrix,
- \( W^{out} : L \times (N + K) \) output weight matrix,
- \( x^n : N \times 1 \) state vector for time step \( n \),
- \( u^n : K \times 1 \) input vector for time step \( n \),
- \( y^n : L \times 1 \) output vector for time step \( n \),
- ‘;’ : vertical vector concatenation,
- \( \tau \) : time constant parameter.

Note that ESNs can have feedback projections from the readout module to the reservoir [Jaeger 2001]. Throughout this chapter, however, we employ ESN architectures without such feedback projections. We choose \( f(x) = \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}, \) which allows for existence of echo states.
6.2. The Echo State Network Model

Figure 6.5: A recurrent sparse random reservoir (left) and a feed-forward sparse random reservoir (right). Numbers denote the neuron indices. A reservoir is guaranteed to be acyclic if connections are allowed only from neurons with lower indices to neurons with higher indices. A recurrent reservoir (left) can be converted into a feed-forward one by inverting the connections that are from higher indexed neurons to lower indexed ones (dashed lines, right).

6.2.2 Feed-Forward Echo State Networks

Acyclic random neural networks with leaky integrator neurons can be used as reservoirs, as they also have fading memory. We refer an ESN with such a reservoir as a Feed-Forward Echo State Network (FF-ESN). Naturally, it can be asked whether FF-ESNs can have the same expressive power as the recurrent ESNs. Intuitively, it can be expected that a recurrent ESN with the same number of neurons discriminates between more input patterns than a FF-ESN, i.e. the recurrent ESN has a much more powerful separation property. In case of adaptive reservoirs, however, it is worthwhile to test whether the reservoir adaptation algorithm works especially better on feed-forward reservoirs than on recurrent ones. Here, we also propose the use of FF-ESNs for functional identification of biological neural networks. A recurrent reservoir can easily be made acyclic by inverting the connections that cause recurrence (figure 6.5).

6.2.3 ESN Learning

Although ESN learning is restricted to readout learning in many cases, there have recently been several approaches to adapt reservoirs, among which a significant improvement over untrained reservoirs was reported by Steil [Steil 2007]. Adopting the intrinsic plasticity rule from real biological systems improved the performance of the backpropagation-decorrelation algorithm.

In the current approach, we adapt the reservoir connectivity and individual neuronal time constants with a one-step propagation of the log-likelihood into the reservoir. This is basically a gradient descent approach, where the log-likelihood of a point process is used as an optimization criterion. We elaborate our approach in section 6.4.
Chapter 6. Functional Modeling of Biological Neural Networks using Reservoir Adaptation for Point Processes

6.3 Point Process Modeling of Spike Data

For detailed background in point process modeling of spike data, we refer to chapter 3. The instantaneous firing rate is defined as a conditional intensity function [Cox 1980, Daley 2003]

\[
\lambda_i(t|I_t, H_t) = \lim_{\Delta \to 0} \frac{P(N_i(t + \Delta) - N_i(t) = 1|I_t, H_t)}{\Delta},
\]

where \(N_i(t)\) is the cumulative spike count of unit \(i\). \(\lambda_i(t|I_t, H_t)\) represents the conditional probability density that a spiking event occurs at unit \(i\). \(I_t\) and \(H_t\) stand for input history and spike history, respectively. For small \(\Delta\), the probability of a spike can be computed by Bernoulli approximation to the point process [Daley 2003, Brown 2003a]

\[
P(\delta N_i(t) = 1|I_t, H_t) \approx \lambda_i(t|I_t, H_t) \Delta,
\]

where \(\delta N_i(t) = N_i(t + \Delta) - N_i(t)\). If \(\lambda\) is a function of \(I_t, H_t\) and parameter set \(\theta\), the log-likelihood \(L\) of a sample count path for a node (electrode) \(i\) is expressed as in [Chornoboy 1988]:

\[
L_i(\theta|N_i) = \log p(N_i|\theta) = \int_0^T \log \lambda_i(t|I_t, H_t, \theta) \, dN_i(t) - \int_0^T \lambda_i(t|I_t, H_t, \theta) \, dt,
\]

where \(p(N_i|\theta)\) is the conditional probability density for the count path. The instantaneous log-likelihood [Brown 2001] of a point process trajectory (spike train) is defined as

\[
\ell_i(t, \theta) = \log \lambda_i(t|I_t, H_t, \theta) \frac{dN_i(t)}{dt} - \lambda_i(t|I_t, H_t, \theta)
\]

Note that

\[
L_i(\theta|N_i) = \int_0^T \ell_i(t, \theta) \, dt.
\]

With a conversion to discrete time, i.e. \(t \to n, \Delta \to 1\), instantaneous log-likelihood becomes,

\[
\ell_i^n = \log \lambda_i^n \delta_{N_i} - \lambda_i^n,
\]

where \(n\) is the discrete time step index. Any parameter \(\theta_j\) of the model is then learned then by the standard gradient descent learning.

\[
\theta_j^{n+1} \leftarrow \theta_j^n + \eta \frac{\partial \ell_i^n}{\partial \theta_j} \bigg|_{\theta_j=\theta_j^n},
\]

where \(\eta\) is the learning rate. For the sake of readability, we will leave the calculation points of partial derivatives out and simply use the notation \(\theta_j^{n+1} \leftarrow \theta_j^n + \eta \frac{\partial \ell_i^n}{\partial \theta_j}\).

In the current work, we model the function, \(\lambda(t|I_t, H_t, \theta)\), by an Echo State Network, which inherently incorporates input and network history into the instantaneous network states.
6.4. Point Process Modeling with ESN and Log Likelihood Propagation

We relate input and reservoir state to the conditional intensity function in discrete time as

$$\lambda(t|I_t, H_t, \theta) \approx \lambda^n = f_{\text{out}}(W^{out} [u^n; x^n]),$$  \hfill (6.3)

where $\lambda^n$ is an $L$-dimensional vector of conditional intensity estimations for time step $n$. We use an exponential $f_{\text{out}}$

$$f_{\text{out}}(\xi) = \exp(A \xi),$$  \hfill (6.4)

where $0 \leq A \leq 1$ is a constant. Exponential functions were already applied in point process modeling and shown to have desirable properties such as avoiding local minima in Generalized Linear Models (GLM) \cite{Paninski2004}.

ESN readout learning can be trivially adapted to point process data, where the output of the ESN estimates the conditional intensity function in batch mode

$$W_{ij}^{out} \leftarrow W_{ij}^{out} + \eta \frac{\partial \mathcal{L}_i}{\partial W_{ij}^{out}},$$  \hfill (6.5)

and also in online mode

$$W_{ij}^{out} \leftarrow W_{ij}^{out} + \eta_{\text{out}} \frac{\partial \ell_n}{\partial W_{ij}^{out}}.$$  \hfill (6.6)

We are going to provide the expansion of the above learning rule and its extension to reservoir adaptation assuming an online mode. Combining 6.2, 6.3, and 6.4 yields

$$\frac{\partial \ell_n}{\partial W_{ij}^{out}} = A x_j^n \delta_{N_i^n} - A x_j^n f_{\text{out}}(\xi^n) = A x_j^n \delta_{N_i^n} - A x_j^n \lambda^n_i,$$  \hfill (6.7)

where $\xi^n = W^{out} [u^n; x^n]$.

6.4.1 Reservoir Adaptation

One interesting question is whether it is sufficient to learn the output weights or whether one needs to adapt the reservoir using the point process log-likelihood as the fitness criterion. In the current work, we adapt recurrent and feed-forward reservoirs with one step propagation of the point process log-likelihood into the reservoir and compare the results to non-adaptive reservoirs.

Any connectivity weight $W_{kl}$ within the reservoir can also be adapted with the gradient descent rule,

$$W_{kl}^{n} \leftarrow W_{kl}^{n-1} + \eta_{\text{res}} \frac{\partial \ell_n}{\partial W_{kl}}.$$  \hfill (6.8)
Chapter 6. Functional Modeling of Biological Neural Networks using Reservoir Adaptation for Point Processes

As we assume the output channels are mutually independent given the parameters,

$$\frac{\partial \ell^n}{\partial W_{kl}} = \sum_{i=1}^{L} \frac{\partial \ell^n_i}{\partial W_{kl}}. \quad (6.9)$$

Obviously, $\ell^n_i$ has a long term dependency on $W_{kl}$ due to reservoir memory, i.e. $\ell^n_i$ not only depends on $W_{kl}^{n-1}$ but also on $W_{kl}^{n-h}$, where $h > 1$. However, for the sake of computational efficiency, and in view of the fact that long term partial derivatives tend to vanish exponentially with respect to time [Bengio 1994], we utilize a one-step propagation of the instantaneous log-likelihood into the reservoir.

$$\frac{\partial \ell^n}{\partial W_{kl}} \approx \frac{\partial \ell^n_i}{\partial W_{n-1}^{kl}}. \quad (6.10)$$

Note that if neuron $\psi$ projects to the readouts, any connection to $\psi$ is updated with one-step propagation. As every reservoir neuron is projecting to the readout units, every reservoir connection is updated with the learning rule.

$$x^n = (1 - \frac{1}{\tau})x^{n-1} + \frac{1}{\tau} f(W^{in} u^n + W^{n-1} x^{n-1}), \quad (6.11)$$

$W_{kl}^{n-1}$ contributes to $\ell^n_i$ for all $i$, $k$ and $l$ through $x^n_k$. We can now utilize the chain rule,

$$\frac{\partial \ell^n_i}{\partial W_{n-1}^{kl}} = \frac{\partial \ell^n_i}{\partial x^n_k} \frac{\partial x^n_k}{\partial W_{n-1}^{kl}}. \quad (6.12)$$

Partial derivatives with respect to state variables can be computed by combining equations 6.2, 6.3, and 6.4.

$$\frac{\partial \ell^n_i}{\partial x^n_k} = A W^{out}_{ik} \delta^n_{N_i} - A W^{out}_{ik} \lambda^n_i. \quad (6.13)$$

Again following equation 6.11,

$$\frac{\partial x^n_k}{\partial W_{kl}} = \alpha_k \frac{\partial f_k(a^n_k)}{\partial a^n_k} x^{n-1}_l. \quad (6.14)$$

where $a^n_k$ is the $k-th$ element of the vector $W^{in} u^n + W^{n-1} x^{n-1}$ and $f_k(a^n_k)$ is the $k-th$ element of the resulting state vector. We define $\alpha_k$ as the inverse of the time constant for reservoir unit $k$, $0 < \alpha_k = 1/\tau_k < 1$. With equations 6.9, 6.10, 6.12, 6.13 and 6.14, the update rule in equation 6.8 is completed.

6.4.1.1 Adapting neuronal time constants

Gradient descent is also used to adapt the time constants of the reservoir neurons. Note that in this case time constant is not a global parameter anymore. Let $\alpha_j^n = 1/\tau_j^n = g(\alpha_j^n) = 1/(1+\exp(\alpha_j^n))$, to have a continuity over all $\alpha_j^n$ and to keep them in the desired range.
6.4. Point Process Modeling with ESN and Log Likelihood Propagation

\[
\frac{\partial \ell^n_i}{\partial \alpha_j^{n-1}} = \frac{\partial \ell^n_i}{\partial x_j^n} \frac{\partial x_j^n}{\partial \alpha_j^{n-1}} \quad (6.15)
\]

\[
\frac{\partial x_j^n}{\partial \alpha_j^{n-1}} = \{-x_j^{n-1} + f_j(W_{\text{inp}} u^n + W x_j^{n-1})\} \frac{dg(\alpha_j^{n-1})}{d\alpha_j^{n-1}} \quad (6.16)
\]

\[
\alpha_j^n \leftarrow \alpha_j^{n-1} + \eta \frac{\partial \ell^n}{\partial \alpha_j^{n-1}}, \quad (6.17)
\]

The complexity of the readout-only learning is \(O(LN)\) for each time step, where \(L\) is the number of readout units. Running the trained ESN on the test input stream has also a complexity of \(O(LN)\) for sparse reservoirs, where the number of connections linearly scales with the number of reservoir units. All the reservoirs we use belong to this sparse type. In this case, the complexity of the one-step log-likelihood propagation is also \(O(LN)\).

6.4.2 Existence of Local Maxima and Confidence Intervals

Due to deep architecture and reservoir transfer function, we cannot guarantee that gradient descent in the reservoir parameters yields uniquely true, i.e. globally optimal, parameters. Point process log-likelihood is not concave with respect to the whole set of parameters. Reservoir adaptation, as a result, is not a tool for finding a globally optimal equivalent of a given BNN. The quality of reservoir adaptation is evaluated based on its improvement on the predictive performance over fixed reservoirs. For a fixed reservoir, on the other hand, learning reduces to readout-only learning, and hence, to adapting the parameters of a Generalized Linear Model \[McCullagh 1989, Paninski 2004\], which maps reservoir states and input to the conditional intensity. In this case, point-process log-likelihood is a concave function of readout parameters (see Appendix B) and does not have non-global local maxima with respect to them \[Paninski 2004\]. This shows that gradient descent in the readout parameters will asymptotically result in a global maximum. The absence of local minima holds for readout parameters under a fixed reservoir or if an adaptive reservoir is fixed after some epochs of training. Note that we take an online (stochastic) gradient descent approach in this work. Although online gradient descent takes a stochastic path in the parameter space for maximization of the log-likelihood, empirical evidence suggests that the average behavior of gradient descent is not affected by online updates \[Bottou 2004\].

Upon training, the observed Fisher Information Matrix can be used to approximate the covariance matrix and confidence intervals on the readout parameters,

\[
\Sigma^{-1}_{\text{out}} = -\nabla_{W_{\text{out}}}^2 \mathcal{L}(W_{\text{out}}|N),
\]

where \(\Sigma_{\text{out}}\) is the estimation of the covariance matrix and \(\nabla_{W_{\text{out}}}^2 \mathcal{L}(W_{\text{out}}|N)\) is the Hessian of the log-likelihood. The Hessian matrix is computed at the final point estimates of the readout parameters. Confidence intervals are computed using the
standard deviations obtained from the diagonal of the covariance matrix. It should be noted that we do not perform a full probabilistic learning of the parameters. We approximate the distribution on the readout parameters by the asymptotic normal distribution of the maximum likelihood estimator [Davison 2003, Pawitan 2001].

6.5 Evaluation of the Learned Models

We compared our results with feed-forward and recurrent architectures to the baseline method where only the readout weights were adapted. For different ESN types and architectures, we comparatively evaluated their capabilities on modeling BNN stimulus-response relations by testing the predictive power of the ESN on the observed spike trains. The continuous ESN output signal was tested for compatibility with the actual observed spikes.

Receiver Operating Characteristic (ROC) curve analysis was employed to test the quality of prediction. ROC curves are extensively used in signal detection theory and machine learning to evaluate prediction performance on binary variables. In a binary classification problem (here spike or no spike), a ROC curve is a plot of true positive rate vs. false positive rate. True positive rate is defined as

\[ TPR = \frac{\# \text{True Positives}}{\# \text{All Positives}} \]

Similarly, false positive rate is defined as

\[ FPR = \frac{\# \text{False Positives}}{\# \text{All Negatives}}. \]

As the ESN output is continuous, true positive rate and false positive rate will vary with respect to a moving threshold. An example ROC curve is shown in figure 6.7 (bottom). In contrast to classification accuracy, the area under a ROC curve (AUC) is more robust with respect to prior class probabilities [Bradley 1997]. An AUC around 0.5 indicates a random prediction whereas an AUC for a perfect prediction will equal 1.0.

6.6 Experimental Results

We employed Echo State Networks with different types and sizes of reservoirs on spike prediction tasks, where the spikes were recorded from simulations of random cortical networks and from cultured networks of cortical neurons. We investigated whether ESNs successfully predict output spikes of BNNs when they are presented with the same input streams.

6.6.1 Simulations of Random Cortical Networks

We simulated 10 surrogate cortical neural networks of 1000 neurons each with 0.1 random connectivity. We used the Izhikevich neuron model and included spike-timing dependent plasticity (STDP) in our simulations [Izhikevich 2006]. At each
Table 6.1: Areas under ROC curves (in %) for the prediction of activity in 10 simulated biological neural networks with fixed recurrent (R-fxd), feed-forward adaptive (FF-adp) and recurrent adaptive (R-adp) reservoirs. • denotes statistical significance for having outperformed the non-adaptive recurrent reservoir of the same size. ∗ denotes a statistical significance for having outperformed the other two reservoir types of the same size (signed-rank test, \( p < 0.05 \)).

<table>
<thead>
<tr>
<th>Res. Size:</th>
<th>Res. Type:</th>
<th>100</th>
<th>500</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R-fxd</td>
<td>FF-adp</td>
<td>R-adp</td>
<td>R-fxd</td>
</tr>
<tr>
<td>BNN1</td>
<td>74.0</td>
<td>78.3</td>
<td>76.8</td>
<td>81.3</td>
</tr>
<tr>
<td>BNN2</td>
<td>76.7</td>
<td>80.6</td>
<td>78.1</td>
<td>84.1</td>
</tr>
<tr>
<td>BNN3</td>
<td>69.4</td>
<td>73.5</td>
<td>70.7</td>
<td>77.0</td>
</tr>
<tr>
<td>BNN4</td>
<td>76.6</td>
<td>78.9</td>
<td>78.0</td>
<td>81.4</td>
</tr>
<tr>
<td>BNN5</td>
<td>74.9</td>
<td>78.0</td>
<td>77.4</td>
<td>81.5</td>
</tr>
<tr>
<td>BNN6</td>
<td>79.0</td>
<td>82.0</td>
<td>81.3</td>
<td>84.8</td>
</tr>
<tr>
<td>BNN7</td>
<td>72.0</td>
<td>74.4</td>
<td>74.1</td>
<td>78.2</td>
</tr>
<tr>
<td>BNN8</td>
<td>71.7</td>
<td>73.0</td>
<td>72.3</td>
<td>76.8</td>
</tr>
<tr>
<td>BNN9</td>
<td>72.3</td>
<td>74.9</td>
<td>74.6</td>
<td>78.5</td>
</tr>
<tr>
<td>BNN10</td>
<td>79.0</td>
<td>80.7</td>
<td>79.0</td>
<td>83.8</td>
</tr>
<tr>
<td>mean</td>
<td>74.6</td>
<td>77.4</td>
<td>76.2</td>
<td>80.7</td>
</tr>
</tbody>
</table>

Figure 6.6: Confidence intervals on the readout parameters of 3 reservoir types of size 100 for a selected biological neural network. The learned readout weights and 99% confidence intervals (±2.576\( \sigma \)) are shown. Confidence intervals are computed by assuming a multivariate normal distribution and an infinite prior covariance matrix. Hence, inverse of the observed Fisher information matrix is used to compute confidence intervals. The weights indexed from 1 to 100 are reservoir-to-output parameters, whereas the weights indexed from 101 to 200 are direct input-to-output parameters. Greater uncertainty for input-to-output parameters are noticeable, which results from low density of the input pulses per channel.
Figure 6.7: Estimated conditional intensity for a selected biological neural network (BNN2) using input history and the output spike history together. Conditional intensity estimations $\lambda$ for all time steps in the testing period are shown in decreasing order (top). A bar is shown if there was indeed a spike observed in the corresponding time step (top). Distributions of conditional intensity, $\lambda$, for time steps with observed spikes and without spikes (middle). By a varying threshold on $\lambda$, true positive rates vs. false positive rates can be calculated (bottom).
one of the 100 input channels was randomly selected from a uniform distribution and a pulse of 5 ms width was sent to the network. Each input channel had excitatory projections to 10% of the neurons in the network. The networks were simulated for 2 hours in real time with STDP and 0.5 hours in real time with frozen synapses thereafter. In each network, the spikes from a randomly selected excitatory neuron were recorded with 5 ms discretization. We then trained ESNs to estimate the conditional intensity of the selected neuron's spiking process from the history of the input pulses.

LI-ESNs with three different reservoir types of various sizes were used for this task, namely recurrent fixed recurrent reservoirs (R-fxd), recurrent adaptive reservoirs (R-adp) and feed-forward adaptive (FF-adp) reservoirs. Reservoir connectivity and each individual time constant were adapted using the method described in section 6.4. For each of the 10 surrogate biological neural networks, we used reservoirs of sizes 100, 500 and 1000. A separate random sparse reservoir for each size and for each surrogate biological network was generated, where each reservoir unit was connected to 10 other units. For each random reservoir, a feed-forward reservoir was generated by inverting the edges that cause recurrence as described in figure 6.5. Note that the same sized feed-forward and the recurrent reservoirs for the same biological neural network had the same topology apart from edge directions. This yielded 60 different reservoirs for the whole experiment.

From each of the 10 BNN simulations we recorded data for 30 minutes in real time. Using a discretization time bin of 5 ms, the simulation yielded time series data for 360,000 time steps, of which 328,000 time steps were used for training and 30,000 time steps were used for testing in each sub-experiment. For the adaptive reservoirs, the training phase included 20 full adaptation iterations of the reservoirs and 60 readout adaptation iterations. One iteration covered one forward traversal of the training data in time. No learning was performed on the test data.

Results of the ROC curve analysis show that the ESN approach performs well in conditional intensity estimation (table 6.1). For all the experiments on 10 different BNNs, the estimated conditional intensity predicts output firing far better than random prediction. For BNN2, point estimates and uncertainty measures of readout parameters after readout-only-training were also presented. (figure 6.6, see section 6.4.2 for computation of uncertainty measures).

We further investigated whether the predictive power of ESNs with feed-forward adaptive reservoirs of leaky integrator neurons is comparable to that of recurrent reservoirs. Adaptive reservoirs are in general better than non-adaptive reservoirs. This difference, however, vanishes with increasing reservoir size (table 6.1). In these experiments, feed-forward adaptive reservoirs significantly outperformed the non-adaptive recurrent reservoirs of all sizes. Furthermore, they significantly outperformed recurrent adaptive reservoirs for sizes of 100 and 500. Although adaptation brought significant improvement also to recurrent reservoirs for sizes 100 and 1000, feed-forward adaptation performed in general better than recurrent adaptation in estimating the conditional intensity. Small reservoirs (i.e. for $N = 100$) of the feed-forward architecture were drastically superior to the recurrent reservoirs.
Figure 6.8: Changes of the training log-likelihood with the number of iterations for reservoirs with different number of neurons (N) and reservoir types (color coded). The performance of the reservoirs is shown for fixed recurrent (blue, dashed), adaptive recurrent (black, continuous) and adaptive feed-forward types (red, *).
6.6. Experimental Results

One can gain some insight on the comparative performances of different reservoirs by looking at the development of the log-likelihood during training (figure 6.8). Note that we adapted the reservoir connectivity and the time constants only in the first 20 iterations. The remaining 60 iterations included only learning of readout parameters. This allowed for observing the effects of two learning phases separately. The log-likelihood clearly increases with fewer iterations in adaptive reservoirs. A sudden drop of training log-likelihood is remarkable at the transition from the full training phase to the readout training phase. Although this transition significantly reduces the fitness and might make the adaptive reservoirs worse than non-adaptive ones, previous full reservoir training pays out in the upcoming iterations.

Note that table 6.1 refers only to the prediction of output spikes from input pulse trains. If the spike history of the output neuron is also taken into account, the prediction performance increases considerably. In this case, the output spike history is also fed into the ESN through an additional input channel. For BNN2, we visualize the conditional intensity estimation that takes also the output spike history into account (figure 6.7). Note the increased area under the ROC curve when compared to the second row in table 6.1. Cross-correlation coefficients around 0 time lag between the conditional intensity estimation and the target spike train also improve when the output spike history is used (figure 6.9), indicating an increased similarity of estimations and observations. Cross-correlation coefficients are computed as

\[
c_{s\lambda}(m) = \frac{1}{T-1} \sum_n [s(n+m) - \bar{s}] [\lambda(n) - \bar{\lambda}],
\]

where \( m \) is the time lag, \( s \) is the binary valued discrete-time signal for the target spike train and \( \lambda(n) \) is the conditional intensity estimation at time step \( n \). \( \bar{s} \) and \( \bar{\lambda} \) stand for mean values, whereas \( \sigma_s \) and \( \sigma_\lambda \) denote standard deviations. \( T \) is the data length in time steps. As for ROC curve areas, the peak around zero lag indicates the similarity between the conditional intensity estimation and the reference spike train.

6.6.1.1 Approximating gradient vectors with one-step propagation

In order to investigate the consequences of one-step approximation to gradient descent, we compared the gradient vectors obtained by full gradient computation and those obtained by one-step approximation. Full gradient computation would replace equations 6.14 and 6.16 as

\[
\frac{\partial x_j^n}{\partial W_{kl}} = (1 - \alpha_j) \frac{\partial x_j^{n-1}}{\partial W_{kl}} + \alpha_j \frac{\partial f_j(a_j^n)}{\partial a_j^n} \left[ \delta_{jk} x_l^{n-1} + \sum_m W_{jm} \frac{\partial x_m^{n-1}}{\partial W_{kl}} \right]
\]

and
Chapter 6. Functional Modeling of Biological Neural Networks using Reservoir Adaptation for Point Processes

Figure 6.9: Cross-correlation coefficients between the conditional intensity estimation and the observed spike patterns for the output neuron of a selected biological neural network (BNN2) with feed-forward ESN adaptation for a reservoir of 1000 neurons. Using only the input stream, AUC = 85.5% (top). Using the input stream and the spike history of the output neurons, AUC = 91.6% (bottom). Each bar shows the correlation coefficient for 1 time step of the ESN (5 ms). Around 0 time lag, the conditional intensity that uses the spike history has a higher correlation with the observed spikes. Note that there is negative correlation for some negative time lags (−90 < timelag < 0). This results from the fact that conditional intensity estimation is very low after an observed spike since the algorithm learns the negative auto-correlation of the output spikes for this time lag interval. In other words, the learned model is reflecting the inter-spike interval in its output.
6.6. Experimental Results

Figure 6.10: Distributions of the correlations (left) and cosines (right) between fully recursive and one-step computations of gradients vectors, $\nabla \ell^n$ and $\nabla^{n-1} \ell^n$, respectively. Results are shown for a selected sparse recurrent reservoir of size 100 and its feed-forward version.

$$\frac{\partial x_j^n}{\partial \alpha_k} = (1 - \alpha_j) \frac{\partial x_j^{n-1}}{\partial \alpha_k} + \alpha_j \frac{\partial f_j(a_j^n)}{\partial a_j^n} \sum_m W_{jm} \frac{\partial x_m^{n-1}}{\partial \alpha_k} + \delta_{jk}[-x_j^{n-1} + f_j(a_j^n)],$$

respectively. The resulting learning rule would be an adaptation of the Real Time Recurrent Learning ([Williams 1989]) for point processes, which is computationally very expensive for our data and reservoir sizes. It is, however, still viable to compare the gradients obtained by one-step propagation to those obtained by full gradient computation for a limited number of time steps. Here, we compared gradient vectors of the log likelihood with respect to all reservoir parameters for 19,000 time steps. We computed the correlation and the cosine between full and one-step gradients for each time step separately,

$$c^n = \frac{1}{R - 1} \sum_r (\frac{\partial \ell^n}{\partial W_r} - \mu_{one}^n)(\frac{\partial \ell^n}{\partial W_r} - \mu_{full}^n)$$

$$\cos(\Omega^n) = \frac{< \nabla^{n-1} \ell^n, \nabla \ell^n >}{|\nabla^{n-1} \ell^n| |\nabla \ell^n|},$$

where $\nabla^{n-1} \ell^n$ and $\nabla \ell^n$ are the gradient vectors at time step $n$ obtained by one-step propagation and by full gradient computation, respectively; $\frac{\partial \ell^n}{\partial W_r}$ and $\frac{\partial \ell^n}{\partial W_r}$ are their $r$-th components; $\mu_{one/full}^n$ and $\sigma_{one/full}^n$ are the mean values and the standard deviations for the components of the gradient vector; $c^n$ and $\Omega^n$ are the correlation and the angle between the full and one-step gradient vectors at time step $n$, respectively.

The distribution of the correlations and cosines provides insight to the consequences of one-step approximation in feed-forward and recurrent reservoirs. Over
19,000 time steps, the correlations between the one-step and full gradient vectors in a selected random reservoir are distributed around a mean value of 0.61 with a standard deviation of 0.034. The mean and standard values are 0.77 and 0.022 for the corresponding feed-forward reservoir (figure 6.10). Cosines constitute a very similar distribution, as the gradient vector components for each time step have approximately a mean value of 0. Strong correlations between the full and one-step gradients deliver additional explanation for the better performance adaptive reservoirs. The fact that the log-likelihood propagates to every reservoir parameter in one-step (due to full reservoir-to-output connectivity) and that one-step gradient is the dominant component of the full gradient information are possible reasons for the detected similarity. Stronger correlations for feed-forward reservoirs are in agreement with the relative better performance of the feed-forward reservoir adaptation.

6.6.1.2 Information encoded in reservoir activity

A canonical correlation analysis of the reservoir activity and the output spiking activity also revealed that output spikes are encoded in the reservoir activity. Prior to readout-only learning, we recorded the reservoir states for one traversal of the training data and performed a canonical correlation analysis (CCA) [Chatfield 2000] with the observed training spikes. We then used parameters obtained by CCA for relating reservoir states to output spikes in the test data and detected maximum correlations for all of the 10 experiments in table 6.1. Although not as good as in readout learning, CCA also reveals the correlation of the reservoir activity with output spikes (figure 6.11), indicating the spike encoding in the reservoir activity.

In addition to CCA of reservoir states and output spikes, we further visualized the memory traces of the input pulses in reservoir states. We compared how past inputs are reflected in the reservoir activity for different reservoir types by considering input triggered averages (ITA). We define the input triggered average for an input-neuron pair as

$$\text{ITA}_{hj}(m) = \frac{\sum_{n=1}^{T} u_h^n x_j^{n+m}}{\sum_{n=1}^{T} u_h^n},$$

where $m$ is the time lag, $u_h^n$ is the input pulse from channel $h$ at time step $n$ (1 or 0), $x_j^{n+m}$ is the state of the $j$-th neuron at time step $n+m$. The normalized ITA is then

$$\hat{\text{ITA}}_{hj}(m) = \frac{\text{ITA}_{hj}(m) - \bar{x}_j}{\sigma_{x_j}}$$

(6.18)

where $\bar{x}_j$ and $\sigma_{x_j}$ are the mean value and the standard deviation for $x_j$, respectively. Figure 6.12 depicts the $\hat{\text{ITA}}_{hj}$ curves for 800 input neuron pairs in 3 different reservoir types for a selected biological neural network. Memory traces for adaptive reservoirs, especially for recurrent adaptive reservoirs, are more scattered over time. This increased heterogeneity in the responses is very likely a reflection of time constants in adaptive reservoirs.
Figure 6.11: Correlation of reservoir activity with the observed spike activity for a selected recurrent reservoir with 100 neurons and its feed-forward version. Canonical correlation analysis (CCA) results also indicate that output spikes are encoded in reservoir activity. Point process readout learning proves more efficient in relating the reservoir activity to output spikes than CCA.
Figure 6.12: Normalized input triggered averages (see equation 6.18) for 20 different input pulses on 40 different reservoir neurons (background, gray). Randomly selected 30 combinations, are highlighted (darker, red) for three different reservoir types. Adaptive reservoirs display more diffused memory traces, i.e. their time lags for the peaks vary more.
6.6. Experimental Results

![Graph showing AUC for different reservoirs](image.png)

Figure 6.13: Areas under ROC curves (AUC) for the event prediction task at the active output electrodes (left) and aggregated AUC for different ratios of training data (right) using fixed recurrent (R-fxd), feed-forward adaptive (FF-adp) and recurrent adaptive (R-adp) reservoirs, all with 500 neurons.

6.6.2 Prediction of Spontaneous Events in Neuronal Cultures

To test our approach on living neural networks, we aimed at predicting spontaneous events in neuronal cultures. We defined an event as a group of spikes recorded by the same MEA electrode, whose maximum inter spike interval is less than 60 ms. The time of the first spike in the group is regarded as the event time. It should be noted again that the activity in neuronal cultures is typically a sequence of network bursts. If there were at least 100 ms between two successive events, we regarded them as parts of different bursts. These criteria mostly excluded isolated spikes from network bursts and clustered fast repetitive firings at an electrode, e.g. a cellular burst, into a single event. We recorded spikes from a neuronal cell culture for 30 minutes, detecting bursts and events according to the above definition. We used the data for training except for the last 200 seconds, which were reserved for testing. We selected approximately 3/4th of the 60 MEA electrodes and treated their events as the input stream; and the remaining electrodes as output. This selection was based on spatial order of the electrodes, i.e. input and output electrodes occupied two distinct areas in the culture. If an electrode had never recorded spikes in the training event train, it was regarded as inactive and was excluded from the learning and testing processes. The evaluation task was to estimate the conditional intensity of the output events for each output electrode at time step \( n \) using the total input event stream until time step \( n \) (1 ms bin size). Note that only very few events occur outside the bursts. Therefore, the algorithm performs learning and prediction only during bursts.

Based on estimates for all time steps during network bursts in the last 200 seconds, we evaluated the predictive performance of the learned ESN models using ROC curve analysis on the target events and estimated conditional intensity. We selected the electrodes that recorded at least 15 events within the evaluation window of 200 seconds.

Figure 6.13 (left) shows the AUC for each electrode using reservoirs of different
types all with 500 neurons. Note that the prediction results in figure 6.13 are based only on input event streams, i.e. the event history of the output electrodes is excluded. Except for output electrode 3, all AUC measures are far above the chance level. This supports the notion that most output neurons in the culture do not fire randomly but they rather follow patterns [Shahaf 2008]. Even electrode 3 is above the chance level but it has low predictability compared to other electrodes. Aggregated test data AUC over active electrodes (except electrode 3) for different ratios of training data is shown in figure 6.13 (right). As expected, AUC improved with more training data. The improvement, however, was not dramatic. For a selected output electrode the details of the ROC analysis is illustrated in figure 6.14.

Table 6.2 shows aggregated areas under ROC curves over all active electrodes except electrode 3 for a selected culture (Culture 1), without (top) and with (middle) using the output event history. Each entry in the table was obtained with a single reservoir of the corresponding size. As expected, the prediction performance usually increased with the reservoir size. Using the event history of the output electrodes did not improve the prediction performance, thus, no additional information could be found in the output event history. Note that utilizing the event history results in an approximately 25% increase in the input dimensionality. Possible redundancy of the event history with the already existing input might have caused a suboptimal performance of the learning algorithm, decreasing the prediction accuracy in several reservoirs. Feed-forward adaptive and recurrent adaptive reservoirs performed better than the fixed reservoirs for most reservoir sizes. As a baseline for comparison purposes, we used the input event rate (regardless of the spatial information, i.e. without using the electrode number) for the conditional intensity estimation (table 6.2, below). Although this method proved to have some predictive power especially for a rate kernel of $20 \text{ms}$, it is obviously not as good as ESN predictors. Cross-correlation analysis reveals further information about comparative performances of different methods (figure 6.15). The higher peak and the smaller width of the cross-correlogram is an indicator for the better performance of recurrent adaptive reservoirs. To further compare different reservoir types, we experimented with 3 different cultures and employed 10 randomly generated reservoirs of size 500. We used the acyclic forms of the same reservoirs for feed-forward adaptation. In 2 of these 3 cultures (1 and 3), the recurrent adaptive reservoirs had the best average predictive power (table 6.3). In the same 2 cultures, the prediction of the recurrent adaptive reservoirs significantly outperformed the others for several output electrodes (table 6.4).

## 6.6.3 Next-Event Prediction in Neuronal Cultures

Recent findings indicate that the temporal order of the events in neuronal cultures carries significant information about the type of stimuli [Shahaf 2008]. In this experimental setting, we investigated whether the ESNs can model the structure in the temporal order of the events. We deleted timing information of the events from the previous setting and, as a result, we obtained data of temporally ordered events.
Figure 6.14: ROC curve analysis of the event timing prediction for active output electrode 1 using a recurrent adaptive reservoir with 500 neurons without using the spike history. Conditional intensity estimations $\lambda$ for all time steps in the testing period are shown in decreasing order (top). A bar for each observed spike in the corresponding time step (top). Distributions of conditional intensity, $\lambda$, for time steps with and without observed spikes (middle). By a varying threshold on $\lambda$, true positive rates vs. false positive rates can be calculated (bottom).
Table 6.2: Areas under ROC curves for the event timing prediction task aggregated over active output electrodes

<table>
<thead>
<tr>
<th>Res. Size</th>
<th>Without History</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>R-fxd 59.9</td>
<td>FF-adp 59.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 63.9</td>
</tr>
<tr>
<td>30</td>
<td>R-fxd 57.9</td>
<td>FF-adp 64.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 68.9</td>
</tr>
<tr>
<td>50</td>
<td>R-fxd 74.3</td>
<td>FF-adp 72.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 69.2</td>
</tr>
<tr>
<td>100</td>
<td>R-fxd 78.9</td>
<td>FF-adp 76.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 76.2</td>
</tr>
<tr>
<td>300</td>
<td>R-fxd 78.3</td>
<td>FF-adp 83.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 82.3</td>
</tr>
<tr>
<td>500</td>
<td>R-fxd 82.7</td>
<td>FF-adp 82.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 85.8</td>
</tr>
<tr>
<td>1000</td>
<td>R-fxd 83.2</td>
<td>FF-adp 83.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 84.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Res. Size</th>
<th>With History</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>R-fxd 57.2</td>
<td>FF-adp 57.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 59.8</td>
</tr>
<tr>
<td>30</td>
<td>R-fxd 62.4</td>
<td>FF-adp 66.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 68.1</td>
</tr>
<tr>
<td>50</td>
<td>R-fxd 69.9</td>
<td>FF-adp 69.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 74.3</td>
</tr>
<tr>
<td>100</td>
<td>R-fxd 76.7</td>
<td>FF-adp 76.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 79.7</td>
</tr>
<tr>
<td>300</td>
<td>R-fxd 79.8</td>
<td>FF-adp 80.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 85.1</td>
</tr>
<tr>
<td>500</td>
<td>R-fxd 82.5</td>
<td>FF-adp 82.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 83.4</td>
</tr>
<tr>
<td>1000</td>
<td>R-fxd 82.1</td>
<td>FF-adp 83.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R-adp 83.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Kernel(ms):</th>
<th>Rate Based Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>AUC: 60.7 64.0 69.3</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
</tr>
<tr>
<td>150</td>
<td></td>
</tr>
<tr>
<td>250</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.3: Mean AUC (%) in the event timing prediction task in 3 different cultures. Results are aggregated over 10 random reservoirs with size 500 for all active output electrodes, for the best active electrode and for the worst active electrode.

<table>
<thead>
<tr>
<th>channel</th>
<th>Culture 1</th>
<th>Culture 2</th>
<th>Culture 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R-fxd</td>
<td>FF-adp</td>
<td>R-adp</td>
</tr>
<tr>
<td>all</td>
<td>81.6</td>
<td>83.1</td>
<td>83.6</td>
</tr>
<tr>
<td>best</td>
<td>84.2</td>
<td>87.0</td>
<td>89.3</td>
</tr>
<tr>
<td>worst</td>
<td>78.8</td>
<td>77.8</td>
<td>76.6</td>
</tr>
</tbody>
</table>
Figure 6.15: Cross-correlation coefficients between the conditional intensity estimation and the observed output events for a selected (1) active output electrode without using the output spike history (res. size of 500). Coefficients are shown for fixed recurrent (a), feed-forward adaptive (b), and recurrent adaptive reservoirs (c). Each bar shows the correlation coefficient for 5 time steps of the ESN (5 ms). As a baseline a prediction based on event rate (without an ESN, conditional intensity proportional to event rate) is also shown (d). The recurrent adaptive reservoir performs better here, considering the height of the peak and the width of the cross-correlogram.

Table 6.4: Pairwise comparison of the reservoir types and methods in the event timing prediction task in 3 different cultures using 10 random reservoirs all with size 500. Each entry in the table denotes the number of electrodes, for which the method in the row significantly outperforms the method in the column (signed-rank test, \( p < 0.05 \)).

<table>
<thead>
<tr>
<th>Culture 1</th>
<th>Culture 2</th>
<th>Culture 3</th>
</tr>
</thead>
<tbody>
<tr>
<td># output el.: 6</td>
<td># output el.: 10</td>
<td># output el.: 9</td>
</tr>
<tr>
<td>R-fxd</td>
<td>FF-adp</td>
<td>R-adp</td>
</tr>
<tr>
<td>R-fxd</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>FF-adp</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>R-adp</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>
Chapter 6. Functional Modeling of Biological Neural Networks using Reservoir Adaptation for Point Processes

Table 6.5: AUC (%) for the next-event prediction task aggregated over all active output electrodes in a selected neuronal culture (1).

<table>
<thead>
<tr>
<th>Res. Size</th>
<th>10</th>
<th>30</th>
<th>50</th>
<th>100</th>
<th>300</th>
<th>500</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-fxd</td>
<td>66.2</td>
<td>71.6</td>
<td>73.7</td>
<td>79.4</td>
<td>82.1</td>
<td>82.9</td>
<td>83.3</td>
</tr>
<tr>
<td>FF-adp</td>
<td>71.3</td>
<td>77.9</td>
<td>79.8</td>
<td>81.5</td>
<td>82.5</td>
<td>83.3</td>
<td>83.1</td>
</tr>
<tr>
<td>R-adp</td>
<td>71.8</td>
<td>72.7</td>
<td>79.4</td>
<td>81.2</td>
<td>82.3</td>
<td>82.4</td>
<td>82.3</td>
</tr>
</tbody>
</table>

Table 6.6: Mean AUC (%) in the next-event prediction task in 3 different cultures. Results are aggregated over 10 random reservoirs with size 300 for all active output electrodes, for the best active electrode and for the worst active electrode.

<table>
<thead>
<tr>
<th>channel</th>
<th>Culture 1</th>
<th></th>
<th>Culture 2</th>
<th></th>
<th>Culture 3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R-fxd</td>
<td>FF-adp</td>
<td>R-adp</td>
<td>R-fxd</td>
<td>FF-adp</td>
<td>R-adp</td>
</tr>
<tr>
<td>all</td>
<td>81.9</td>
<td>82.6</td>
<td>82.2</td>
<td>75.2</td>
<td>75.7</td>
<td>75.4</td>
</tr>
<tr>
<td>best</td>
<td>93.1</td>
<td>93.4</td>
<td>93.2</td>
<td>88</td>
<td>89.1</td>
<td>88.5</td>
</tr>
<tr>
<td>worst</td>
<td>54.7</td>
<td>56.3</td>
<td>55.2</td>
<td>62.4</td>
<td>60.4</td>
<td>62.5</td>
</tr>
</tbody>
</table>

More precisely, we used operational time steps for ESN analysis, i.e. each event appeared only in 1 time step and each time step contained only 1 event.

In this setting, we treated every electrode as output and used the whole event history for prediction. The task was again to estimate the conditional intensity of the output events for each electrode at time step $n$ using the total event stream until time step $n$. Note that point process modeling is not necessarily the optimal tool for this data, where each abstract time step can contain only 1 event. The general framework taken in the current work, however, can still be employed for predictive modeling of temporal event orders. Again, ROC curve analysis indicates a good average prediction quality with larger reservoirs, feed-forward adaptive reservoirs outperforming the recurrent reservoirs up to sizes of 500, (table 5, figure 6.16).

To further compare different reservoir types, 10 random reservoirs of 300 neurons together with their acyclic forms were tested on 3 different cultures (table 6.6). The predictability varied with respect to the electrode that recorded the event, with AUC ranging from 50.5% to 93.4%. In pairwise comparison of the 3 reservoir types, FF-adp reservoirs significantly (signed-rank test, $p < 0.05$) outperformed other methods for many electrodes (table 6.7). Conversely, the number of electrodes, for which the other methods performed better, was much smaller.
Table 6.7: Pairwise comparison of the reservoir types and methods in the next-event prediction task in 3 different cultures using 10 random reservoirs all with size 300. Each entry in the table denotes the number of electrodes, for which the method in the row significantly outperforms the method in the column (signed-rank test, \( p < 0.05 \)).

<table>
<thead>
<tr>
<th></th>
<th>Culture 1 # output el.: 26</th>
<th>Culture 2 # output el.: 34</th>
<th>Culture 3 # output el.: 38</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R-fxd FF-adp R-adp</td>
<td>R-fxd FF-adp R-adp</td>
<td>R-fxd FF-adp R-adp</td>
</tr>
<tr>
<td>R-fxd</td>
<td>0 5 5</td>
<td>0 6 5</td>
<td>0 6 11</td>
</tr>
<tr>
<td>FF-adp</td>
<td>11 0 8</td>
<td>22 0 20</td>
<td>12 0 17</td>
</tr>
<tr>
<td>R-adp</td>
<td>7 1 0</td>
<td>6 1 0</td>
<td>7 0 0</td>
</tr>
</tbody>
</table>

Figure 6.16: Estimated conditional intensity for a selected output electrode (5) of the neuronal culture in the next-event-prediction task using a feed-forward adaptive reservoir with 500 neurons. Conditional intensity estimations \( \lambda \) for all time steps in the testing period are shown in decreasing order (top). A bar is shown if there was indeed a spike observed in the corresponding time step (top). Distributions of of conditional intensity, \( \lambda \), for time steps where there was an observed spike and where there was no spike (middle). By a varying threshold on \( \lambda \), true positive rates vs. false positive rates can be calculated (bottom).
6.7 Conclusion

Our results indicate that reservoir computing is a potential candidate for modeling neural activity including neural encoding and decoding. With their expressive power for different activity measures (e.g., spike rates, correlations etc.), reservoir computing tools might help for analysis of neural data. In our experiments, ESNs of leaky integrator neurons proved successful for modeling response-dynamics (e.g., stimulus-response relations and spatio-temporal dynamics) of simulated and cultured biological neural networks.

On the methodological side, we showed that ESN learning algorithms can be adapted for event data, such as spikes or spike groups, using a point process framework. We proposed a reservoir adaptation method for event data, which can be used to adapt connectivity and time constants of the reservoir neurons. The experimental results indicate that reservoir adaptation can significantly improve the ESN performance over readout-only training.

We utilized feed-forward networks with leaky-integrator neurons as reservoirs with a comparable predictive power to recurrent reservoirs when trained with log-likelihood propagation. In modeling stimulus-response relations of simulated BNNs, feed-forward reservoir adaptation outperformed other methods up to 500 neurons. This outperformance was statistically significant. For the event timing prediction task in neuronal cultures, adaptive recurrent reservoir adaptation outperformed the other methods (in 2 of 3 cultures), whereas feed-forward adaptation were better in the next-event type prediction task in all 3 cultures. This might indicate that the type of encoding in neural systems (order or timing) favors different architectures for decoding. An analysis of the structure-coding relations, however, is beyond the scope of this note. Why feed-forward reservoir adaptation can work better than recurrent reservoir adaptation necessitates also more theoretical analysis. We manually optimized global parameters (spectral radius, learning rates and A) for recurrent fixed reservoirs. Recurrent adaptive reservoirs started from these values. Although we can think of no obvious disadvantage for recurrent reservoirs, feed-forward adaptation outperformed recurrent adaptation in some tasks. We experimentally showed that one-step propagation approximates the gradients better in feed-forward reservoirs than in recurrent ones. In our opinion, better structuring of the reservoir parameters, i.e. separation of the memory parameters from the reservoir connectivity, might further underlie the relative better performance of feed-forward adaptation. For instance, a small connectivity change in the recurrent adaptation might have a more dramatic effect on the reservoir memory than in case of feed-forward adaptation. Our findings on feed-forward networks are also in accordance with the recent work by Goldman and Ganguli et al. [Ganguli 2008, Goldman 2009, Murphy 2009], who show that stable fading memory can be realized by feed-forward or functionally feed-forward networks and that feed-forward networks have desirable properties in terms of trade-off between noise amplification and memory capacity.
Activity in neuronal cultures are composed of network bursts that are self-organized into temporal activity patterns, i.e. dynamic attractors. In this chapter, we present a reservoir computing approach for mimicking these attractors of biological neural networks. We employ an Echo State Network (ESN) with leaky integrator neurons as a modeling tool. A simple readout training approach yields predictive model of the temporal activity pattern in the global firing rate. The initial state at the pattern onset is used as a cue and the attractor is simulated by the model. The simulated attractor displays a high correlation with the actual one observed in the culture. High correlation measures and successful prediction of attractor-types indicate a good modeling quality. The developed model can also be used for visualizing the underlying structure in the recorded raw data. Investigating the model dynamics reveals multiple types of attractor states in the system.
7.1 Introduction

Brain activity displays spatio-temporal activity patterns that have experimentally been shown to encode function and information in the cortex [Abeles 1993, Lindsey 1997, Mainen 1995, Prut 1998, Riehle 1997, Villa 1999, Stopfer 2003]. These spatio-temporal patterns have also been investigated in terms of attractor systems [Amit 1992, Barak 2007, Rabinovich 2001, Leonardo 2005, Bibitchkov 2002, Romani 2006]. In other words, the patterns are claimed to be persistent states to which neural systems tend to evolve upon functionally relevant changes in the activity dynamics. For instance, attractor dynamics have been proposed to be relevant to motor and cognitive functions [Stopfer 2003, Miller 1996, Barak 2007].

The prevalence of spatio-temporal activity patterns is not limited to in vivo networks. In cultured networks of neo-natal rates, oscillatory activity has been shown to have complex patterns in their local spike activity [Rolston 2007, Wagenaar 2006b], aggregated firing rates [Wagenaar 2006a, Wagenaar 2006b], and spatial spread of reverberations [Shahaf 2008]. These findings indicate that such patterns do not require specific cortical structure for displaying such patterns. Even in in vitro networks with random topologies, reverberations self-organize into precise temporal patterns, i.e. dynamic attractors [Wagenaar 2006a]. Wagenaar et al. have demonstrated that the activity in such cultures are self-organized into dynamic attractors. In other words, cultures form bursts of emergent precise temporal patterns as a result of their spontaneous fluctuations. The description of a temporal pattern there is based on the aggregated firing rate in the culture over the whole recording electrodes. It is also shown in the same work that temporal structure in the global firing rates can be grouped into several clusters, each of which can be seen as a separate burst type. Figure 7.1 shows an example with 3 different burst types, i.e. attractor states, from a selected neuronal culture.

A good model for biological neural systems should be able to display and simulate comparable patterns in order to reveal the information encoded in such patterns. Can the model of a neural system successfully mimic the temporal patterns under the same input and/or initial state? We claim that this is the minimal requirement for a network-level model, in order to link the dynamics and function. Our approach for this task is to employ a generic network model, which can be assumed to be universal for problems of neural encoding. The generic network model should display a rich repertoire of complex spatio-temporal patterns, which could then be matched to the observed biological patterns by parameter tuning.

7.1.1 Problem Specification

In the current chapter, we present a method to model persistent temporal patterns, i.e. attractor states, of biological neural networks. Here, we presume that attractor states have important functional relevance in biological neural networks. We employ dissociated cortical cultures as an example of random biological neural networks without specific connectivity and simulate the corresponding burst types based on
7.2. The ESN Model

Figure 7.1: 3 different types of dynamic attractors from a selected culture. The thick dashed red line denotes the average pattern. The individual global firing rate patterns are shown with light green in the background.

Figure 7.2: Architecture of Echo State Networks

a cue signal. The cue signal is basically an indicator for the dynamic state of the neuronal culture. In our experiments, the cue signal is composed of a snapshot of the individual firing rates at each electrode for one time step at the beginning of the burst onset. More specifically, a vector including \( C \) firing rates at \( C \) different electrodes is passed to our model and the model simulates temporal pattern that is intended to match the observed pattern in the culture.

This task specification poses a test for artificial networks’ capacity to model dynamic state trajectory in the culture. As the information is primarily encoded in the spatio-temporal network dynamics, we believe that a neural modeling tool should be able to deal with dynamic trajectories.
7.2 The ESN Model

We employ the following LI-ESN model:

\[ x^{n+1} = (1 - \frac{1}{\tau}) x^n + \frac{1}{\tau} f(W^{in} u^{n+1} + W x^n), \quad \tau \geq 1. \]

\[ y^n = f^{out}(W^{out} [u^n; x^n]), \]

where,

- \( W \): \( N \times N \) internal reservoir weight matrix,
- \( W^{in} \): \( N \times K \) input weight matrix,
- \( W^{out} \): \( L \times (N + K) \) output weight matrix,
- \( x^n \): \( N \times 1 \) state vector for time step \( n \),
- \( u^n \): \( K \times 1 \) input vector for time step \( n \),
- \( y^n \): \( L \times 1 \) output vector for time step \( n \),
- \('\cdot'\): vertical vector concatenation,
- \( \tau \): time constant global parameter.

We choose \( f(x) = \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \).

Note that contrast to the case in chapter 6, we feedback from the readout module to the input. This projection is active when we use the ESN for testing. During training, this feedback input is replaced by the actual teacher signal (teacher forcing [Jaeger 2001]).

7.2.1 ESN Learning

Here, we utilize Wiener-Hopf method for readout learning [Jaeger 2007a]. Let \([X; U] = X\),

\[ \bar{W}^{out} = p R_x^{-1}, \]

where \( R_x^{-1} = XX^T \) and \( p = YX^T \).

7.3 Methods for Modeling and Evaluation of the Learned Models

7.3.1 Modeling

Recorded spike trains via MEAs were analyzed for this modeling task. It should be noted again that the activity in neuronal cultures is typically a sequence of network bursts. Network bursts were detected. If there were at least \( 45 \text{ ms} \) between two successive spikes, we regarded them as parts of different bursts. The bursts with the involvement of fewer than 5 electrodes or fewer than 15 spikes were left out.
of the experiment. We manually confirmed that this scheme of burst detection did not leave any significant activity out. The global, i.e. culture-wide, individual firing rates were computed using a rectangular kernel of 60 ms width and a 10 ms sampling period. In order to reduce the computational cost of the analysis, the data is squeezed to operational time steps, where the recordings out of bursts were left out and bursts are concatenated to each other with a 20 ms (2 time steps) of 0 signal in between. A vector of length \( C + 1 \), containing the firing rates for each of the \( C \) channels as well as the global firing rate, was snapshot for one time step at the beginning of the burst. This vector was used as a cue signal for simulating the rest of the burst by the ESN model. The time step for snapshotting the cue signal was manually set to the value, where the global firing rate exceeded 60% of its average level. This usually corresponded to the first or to the second time step (10 ms) at the beginning of the burst. The ESN model was reset and presented with the cue signal for each of the bursts. The model then simulated the rest of the burst in terms of global firing rate. Note that the model did not get any information from the culture apart from the cue signal. It, as a dynamical system, rather used its own output in a feedback loop, in order to model the fine details of the attractor, towards which the culture is drawn.

### 7.3.2 Evaluation

We compared the temporal patterns of global firing rates in the culture to those of the ESN model. The correlation coefficient between the concatenated predicted patterns and the concatenated corresponding observed patterns was used as a measure of similarity. The correlation coefficient between the observed and predicted signals is

\[
c_{dy} = \frac{1}{T-1} \sum_{n}[d(n) - \bar{d}][y(n) - \bar{y}] / \sigma_d \sigma_y,
\]

where \( n \) is a time step index, \( d(n) \) is the observed global firing rate signal in the culture and \( y(n) \) is the predicted time signal at time step \( n \). \( \bar{s} \) and \( \bar{y} \) stand for mean values, whereas \( \sigma_s \) and \( \sigma_y \) denote standard deviations. \( T \) is the data length in time steps.

We also computed the normalized root mean squared error between the observed and the predicted signal as

\[
NMRSE_{dy} = \sqrt{\frac{\sum_{n}[d(n) - y(n)]^2}{[\max(d) - \min(d)]}}.
\]

#### 7.3.2.1 Types of temporal burst patterns and classification

Though correlation and NMRSE values are indicators a similarity between observed and predicted signals, they do not suffice to demonstrate how the types of observed temporal patterns, i.e. attractors, match of those of predicted ones. In order to test the type match between the pattern types, we classified the temporal burst patterns
into several types (3 in all the cultures) using manually written rules. The rules contained criteria about the timing and number of the peaks in the patterns. Note that the same rules applied to observed and predicted patterns. As the rules varied for each individual culture, we give the details of pattern classification in section 7.4.

Upon manual classification of the burst patterns, we evaluated the type match between the observed and the predicted patterns by computing precision and recall measures for each type separately. For every possible type $c$,

$$\text{Precision}(c) = \frac{\# \text{Correctly predicted as } c}{\# \text{Predicted as } c}.$$  

$$\text{Recall}(c) = \frac{\# \text{Correctly predicted as } c}{\# \text{Actually belong to } c}.$$

Precision and recall measures further indicated that how well the model followed the culture in falling to one of the possible several attractors.

### 7.4 Experimental Results

We employed ESNs to model the dynamic attractors in 3 different neuronal cultures. We investigated how similar the trajectories ESN models were to those of cultures in terms of global firing rate. Computation of firing rates, burst detection, cue signal detection and evaluation of similarity were performed as described in section 7.3.

#### 7.4.1 Culture 1

##### 7.4.1.1 Preparation and pharmacological treatment

Cells were derived from cortices of neonatal rats (wistar strain) by mechanical (chopping with scalpel, trituration) and enzymatical (0.05 %, Trypsin, 15 minutes at 37°C, Sigma) dissociation and plated at densities of 0.5 - 1.0 Mio cells per cm$^2$ onto polyethyleneimine-coated micro-electrode arrays (59 TiN electrodes, 200/500 µm electrode pitch, Multi Channel Systems). Cultures developed in 1ml growth medium, minimum essential medium (Gibco) supplemented with heat-inactivated horse serum (5%, Sigma), L-glutamine (0.5mM, Sigma), glucose (20 mM, Sigma) and gentamicin (2 µg/ml, Sigma). One third of medium was exchanged twice per week. Cultures were maintained at 5% CO$_2$ and 37°C.

The culture developed under chronic treatment with the PKC inhibitor Goe6976 (1 µM, Calbiochem) that leads to structurally altered networks with differences in the bursting dynamics.

##### 7.4.1.2 Recordings and results

59 active channels were involved in the recordings, which spanned 137.78 minutes. 203 burst were detected in the channels. After deleting the silent periods outside of the bursts, computation of the rates and sampling the firing rates with a sampling
Figure 7.3: Comparison of the observed firing rate (solid, blue) and the predicted firing rate (dashed, red) in culture 1. Light blue shaded regions in the background indicate the intervals, where prediction is done based on the cue signal. Cue signal is the spatial pattern containing the firing rates of all electrodes at just 1 time step (10 ms) before the shaded region. Over the combined prediction intervals, the overall correlation coefficient between the predicted and the observed signal is 0.88. The labels above the bursts denote the burst types resulting from categorizing rules. Note that 1 out of the 43 bursts does not belong to any type according to the categorizing rules.

period of 10 ms, the data contained 9,887 time steps. The mean average global firing rate of the preprocessed data was 2.71 kHz. The cue signal was taken as a snapshot of the firing rates at one time step at the beginning of the burst, where the global firing rate exceeded 60% of its average value, equalling 1.63 kHz. 72% of the time steps were used for training and 27% were used for testing purposes. The initial 1% was left out from the analysis. Note that the feedback loop of the ESN model was only functional in the test phase, whereas the feedback signal was replaced by the observed target signal in the training phase (teacher forcing). A randomly generated ESN reservoir with 350 neurons and 0.1 connectivity was used for modeling. The training involved readout training of reservoir-to-output and input-to-output parameters and employed Wiener-Hopf learning rule. The observed firing rate and the predicted firing rate displayed high similarity 7.3.

As depicted in figure 7.3, the temporal activity patterns in the culture differed from each other in terms of timing of the first peak and number of the peaks. We manually wrote some rules to categorize the bursts into different types. The rules are shown in table 7.1, which produced 3 different burst types (figure 7.4). Once we classified each burst into one of the possible types, we then evaluated how the types of the simulated, i.e. predicted, bursts match to those of observed bursts. A precision/recall analysis revealed that the types from the culture and the model fit
if (First Peak Time < 12 time steps) and (Peak Value > Peak Threshold) and (Peak Count = 1)

    then Burst Type ← 1

else if (First Peak Time ≥ 12 time steps) and (Peak Value > Peak Threshold) and (Peak Count = 1)

    then Burst Type ← 2

else if (First Peak Time ≥ 12 time steps) and (Peak Value > Peak Threshold) and (Peak Count ≥ 2)

    then Burst Type ← 3

else Burst Type ← Undefined

Table 7.1: Rules for categorizing the bursts into types in culture 1.
7.4. Experimental Results

Figure 7.4: 3 different types of dynamic attractors in culture 1. The thick dashed red line denotes the average pattern. The individual global firing rate patterns are shown with light green in the background. Note that type 2 occurs only twice in this data.

perfectly (table 7.2). As type 2 and type 3 are close to each other, our investigation involved the precision/recall analysis, in case where those two types are merged to a single type (table 7.2).

Note that results presented in table 7.2 and figure 7.3 were produced one randomly generated reservoir. Another randomly generated reservoir of the same size produced the results in figure 7.5 and table 7.3.

In order to assess the variability of the modeling power of random reservoir ESNs, we generated 10 random reservoirs with 350 neurons each. Upon readout training, we computed the mean values and standard values of several predictive performance measures (figure 7.6). The correlation coefficients and the NMRSE values indicated good prediction quality with low standard deviation. Precision/recall values had also low standard deviations for major types, whereas they had higher standard deviations for underrepresented types.

7.4.1.3 Information encoded in reservoir activity

The ESN model was successful in predicting the dynamic trajectory and relating the cue signal to the type of the trajectory that culture takes. Here, we also show that the model itself is a tool for visualizing the structure in the data. The trained ESN model also becomes susceptible to the underlying structure of the target signal. Figure 7.7 (left column) shows the activity levels of 3 randomly selected neuron pairs, plotted one neuron versus the other one for each pair. The plots clearly reflect the structure in the culture activity. Each reservoir neuron becomes responsive the different features in the cue signal. Their second order joint activities follow several different trajectories. If these trajectories are colored according to the manual classification of the bursts, the clear match between manually defined burst types and the trajectories of the pair activities is observed. Contrarily, such distinct trajecto-
Chapter 7. Modeling Dynamic Attractor States of Biological Neural Networks using Reservoir Computing

Figure 7.5: Comparison of the observed firing rate (solid, blue) and the predicted firing rate (dashed, red) in culture 1. Light blue shaded regions in the background indicate the intervals, where prediction is done based on the cue signal. Cue signal is the spatial pattern containing the firing rates of all electrodes at just 1 time step (10 ms) before the shaded region. Over the combined prediction intervals, the overall correlation coefficient between the predicted and the observed signal is 0.88. The labels above the bursts denote the burst types resulting from categorizing rules. Note that 1 out of the 43 bursts does not belong to any type according to the categorizing rules.
7.4. Experimental Results

Table 7.2: Confusion matrix and precision/recall measures for 3-class (top) and 2-class (bottom) prediction tasks on culture 1 using a selected ESN of size 350. Note that 1 out of the 43 bursts does not belong to any type according to the categorizing rules.

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type 1</td>
<td>35</td>
<td>0</td>
<td>0</td>
<td>1.00</td>
</tr>
<tr>
<td>actual type 2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1.00</td>
</tr>
<tr>
<td>actual type 3</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
<th>recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type 1</td>
<td>35</td>
<td>0</td>
<td>1.00</td>
</tr>
<tr>
<td>actual type 2</td>
<td>0</td>
<td>7</td>
<td>1.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
</tr>
</tbody>
</table>

ries reflecting the underlying structure in the data are not noticeable in comparative plots of individual channel rates (figure 7.7, right column).

As the reservoir size is greater than the number of electrodes, rates from the individual channels as a one-time-step-cue signal is projected to a higher dimensional space. The new representation space accommodates better separability. This facilitates the better separation of cue signals in the reservoir. Not only the cue signal, but also the feedback signal is more distinctly represented in the reservoir. As a result, trajectories of the attractor states follow different distinguishable paths in reservoir dynamics. Note that inter-reservoir connectivity is randomly set and fixed as well as input-to-reservoir connectivity. In other words, the projection of the cue signal to the reservoir is random and not affected by training.

7.4.2 Culture 2

7.4.2.1 Preparation and pharmacological treatment

The culture was prepared with the same procedure as with culture 1. Analyzed data includes recordings from acute drug application experiment with Bicuculline (1 µM, Sigma) and Ifenprodil (10 µ M, Sigma), which alter burst structure and frequency.
Table 7.3: Confusion matrix and precision/recall measures for 3-class (top) and 2-class (bottom) prediction tasks on culture 1 using a selected ESN of size 350. Note that 1 out of the 43 bursts does not belong to any type according to the categorizing rules.

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type</td>
<td>1</td>
<td>35</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>recall</td>
<td>1.00</td>
<td>1.00</td>
<td>0.83</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
<th>precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type</td>
<td>1</td>
<td>35</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>recall</td>
<td>1.00</td>
<td>1.00</td>
<td></td>
</tr>
</tbody>
</table>

7.4.2.2 Recordings and results

39 active channels were involved in the recordings, which spanned 127.1 minutes. 1173 burst were detected in the channels. After deleting the silent periods outside of the bursts, computation of the rates and sampling the firing rates with a sampling period of 10 ms, the data contained 22,137 time steps. The mean average global firing rate of the preprocessed data was 0.23 kHz. The cue signal was taken as a snapshot of the firing rates at one time step at the beginning of the burst, where the global firing rate exceeded 60% of its average value, equalling 0.14 kHz. 72% of the time steps were used for training and 27% were used for testing purposes. The initial 1% was left out from the analysis. Target (teacher) signal was forced during training. Randomly generated ESN reservoirs with 550 neurons and 0.1 connectivity for each is used for modeling. The training involved again readout training of reservoir-to-output and input-to-output parameters and employed Wiener-Hopf learning rule. The observed firing rate and the predicted firing rate displayed high similarity.

Figure 7.8 illustrates that the temporal activity patterns in the culture, which differed from each other in terms number of peaks. Also for this culture, we manually wrote some rules to categorize the bursts into different types (table 7.4), which produced 3 different burst types (figure 7.9). We evaluated how the types of the simulated, i.e. predicted, bursts match to those of observed bursts. A precision/recall analysis revealed that the types from the culture and the model fit quite well also in
7.4. Experimental Results

Figure 7.6: Mean values and standard deviations of prediction quality measures for culture 1 generated by 10 randomly generated and readout-trained reservoirs with 350 neurons each.

if (Peak Value > Peak Threshold) and (Peak Count = 1)

then Burst Type ← 1

else if (Peak Value > Peak Threshold) and (Peak Count = 2)

then Burst Type ← 2

else if (Peak Value > Peak Threshold) and (Peak Count > 2)

then Burst Type ← 3

else Burst Type ← Undefined

Table 7.4: Rules for categorizing the bursts into types in culture 2.
Figure 7.7: Reflection of the structure in the global firing rate in the reservoir activity. Activity of a single neuron versus that of another is plotted for 3 randomly selected pairs (left). The gray trajectory in the background show the jump pair activity state between the end of the burst and by-cue-generated initial state for the next burst. The rate recorded by a MEA electrode versus that of another is plotted for 3 randomly selected electrode pairs (right). Trajectories are highlighted in blue (type 1), red (type 2) and magenta (type 3) in order to show the match between our manual classification and the pair activity in both group of plots.
Figure 7.8: Comparison of the observed firing rate (solid, blue) and the predicted firing rate (dashed, red) for a selected time interval in culture 2. Light blue shaded regions in the background indicate the intervals, where prediction is done based on the cue signal. Cue signal is the spatial pattern containing the firing rates of all electrodes at just 1 time step before the shaded region. Over the combined prediction intervals, the overall correlation coefficient between the predicted and the observed signal is 0.82.
Chapter 7. Modeling Dynamic Attractor States of Biological Neural Networks using Reservoir Computing

Figure 7.9: 3 different types of dynamic attractors in culture 2. The thick dashed red line denotes the average pattern. The individual global firing rate patterns are shown with light green in the background.

this case, especially, where type 2 and type 3 are merged to a single type (table 7.5). Statistics on the predictive performance measures over 10 randomly generated reservoirs with the same size show that the prediction quality is not limited to a single ESN model (figure 7.10).

7.4.3 Culture 3

7.4.3.1 Preparation and pharmacological treatment

The culture was prepared with the same procedure as with culture 1. There was no pharmacological treatment to this culture.

7.4.3.2 Recordings and results

59 active channels were involved in the recordings, which spanned 100.4 minutes. 1292 burst were detected in the channels. After deleting the silent periods outside of the bursts, computation of the rates and sampling the firing rates with a sampling period of 10 ms, the data contained 38,442 time steps. The mean average global firing rate of the preprocessed data was 0.36 kHz. The cue signal was taken as a snapshot of the firing rates at one time step at the beginning of the burst, where the global firing rate exceeded 60% of its average value, equalling 0.21 kHz. 72% of the time steps were used for training and 27% were used for testing purposes. The initial 1% was left out from the analysis. Target (teacher) signal was forced during training. Randomly generated ESN reservoirs with 700 neurons and 0.1 connectivity for each is used for modeling. The training involved again readout training of reservoir-to-output and input-to-output parameters and employed Wiener-Hopf learning rule. The observed firing rate and the predicted firing rate displayed high similarity 7.11.

For this culture, we again used manual rules to categorize the bursts into different types (table 7.6) based on number of peaks, which produced 3 different burst types
Table 7.5: Confusion matrix and precision/recall measures for 3-class (top) and 2-class (bottom) prediction tasks on culture 2 using a selected ESN of size 550. (*)8 of the predicted bursts could not be fit into a type.

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>257</td>
<td>18</td>
<td>0</td>
<td>0.91*</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>52</td>
<td>1</td>
<td>0.91 recall</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1.00</td>
</tr>
</tbody>
</table>

| precision      | 0.98 | 0.74 | 0.75 |

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>257</td>
<td>18</td>
<td></td>
<td>0.91 recall</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>56</td>
<td></td>
<td>0.93</td>
</tr>
</tbody>
</table>

| precision      | 0.98 | 0.76 |      |

Figure 7.10: Mean values and standard deviations of prediction quality measures for culture 2 generated by 10 randomly generated and readout-trained reservoirs with 550 neurons each.
Figure 7.11: Comparison of the observed firing rate (solid, blue) and the predicted firing rate (dashed, red) for a selected time interval in culture 3. Light blue shaded regions in the background indicate the intervals, where prediction is done based on the cue signal. Cue signal is the spatial pattern containing the firing rates of all electrodes at just 1 time step before the shaded region. Over the combined prediction intervals, the overall correlation coefficient between the predicted and the observed signal is 0.69.
if (Peak Value > Peak Threshold) and (Peak Count = 1)

then Burst Type ← 1

else if (Peak Value > Peak Threshold) and (Peak Count = 2)

then Burst Type ← 2

else if (Peak Value > Peak Threshold) and (Peak Count > 2)

then Burst Type ← 3

else Burst Type ← Undefined

Table 7.6: Rules for categorizing the bursts into types in culture 3.

in figure 7.12. We repeated the precision/recall analysis, which revealed that the types from the culture and the model fit quite well also in this case, especially, where type 2 and type 3 are merged to a single type (table 7.7). Statistics on the predictive performance measures over 10 randomly generated reservoirs with the same size were shown in figure 7.13. The prediction quality is also not limited to a single ESN model in this case.

7.5 Conclusion

Our findings indicate that a random recurrent sparse network can model the attractor states of biological neural networks with simple readout training. We demonstrated that ESN models can follow similar temporal patterns to those in neuronal cultures. The type of trajectory that the model takes also matches the type of the trajectory that the biological neural network takes. In this work, we showed that internal cues can be related to persistent temporal patterns, i.e. attractor states. With their expressive power for representing various features of the internal/external cues, ESN models extends neural encoding to state trajectories rather than steady state activity parameters. Reservoir’s separation property allow for following different output trajectories upon initialization with different cue signals. We furthermore showed that reservoir dynamics reflects possible structures in the observed neural activity by visualizing the joint trajectories of randomly selected reservoir-unit pairs.
Chapter 7. Modeling Dynamic Attractor States of Biological Neural Networks using Reservoir Computing

Figure 7.12: 3 different types of dynamic attractors in culture 3. The thick dashed red line denotes the average pattern. The individual global firing rate patterns are shown with light green in the background. Note that type 3 presumably contains more than 1 subtypes that cannot be distinguished with our classification rules.

Table 7.7: Confusion matrix and precision/recall measures for 3-class (top) and 2-class (bottom) prediction tasks on culture 1 using a selected ESN of size 700.

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>203</td>
<td>31</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>59</td>
<td>64</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>precision</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.77</td>
<td>0.64</td>
<td>0.95</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>predicted type</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>203</td>
<td>31</td>
</tr>
<tr>
<td>2</td>
<td>60</td>
<td>90</td>
</tr>
<tr>
<td>precision</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.77</td>
<td>0.74</td>
<td></td>
</tr>
</tbody>
</table>
Figure 7.13: Mean values and standard deviations of prediction quality measures for culture 3 generated by 10 randomly generated and readout-trained reservoirs with 700 neurons each.
Chapter 8

An Adaptive Computing Method for Mining the Structure of Biological Neural Networks: NeuronRank

Contents

8.1 Introduction ........................................... 104
  8.1.1 Link mining and networked data ..................... 104
  8.1.2 Neuronal networks of the mammalian cortex .......... 105
  8.1.3 Motivation and scope of this chapter ............... 106
8.2 Overview of the method .................................. 108
8.3 The network models ..................................... 109
  8.3.1 Neuron model ..................................... 109
  8.3.2 Types of networks ................................ 110
8.4 Activity-related features ................................ 110
8.5 The NeuronRank Algorithm .............................. 111
8.6 Structural features ..................................... 112
  8.6.1 NeuronRank features ............................... 112
  8.6.2 Network motifs ................................... 112
  8.6.3 Cluster coefficient ................................ 113
8.7 Experimental results ................................. 113
8.8 Conclusions .............................................. 118

Until the current point of this thesis, we focused on the application of adaptive computing tools to functional modeling of BNNs. In this chapter, we extend the application focus and show that adaptive computing tools can also be used in mining the structure of BNNs.

Because it is too difficult to relate the structure of a cortical neural network to its dynamic activity analytically, we employ machine learning and data mining to learn structure-activity relations from sample random recurrent cortical networks and corresponding simulations. Inspired by the PageRank and the Hubs & Authorities algorithms for networked data, we introduce the NeuronRank algorithm,
which assigns a source value and a sink value to each neuron in the network. Source
and sink values are used as structural features for predicting the activity dynamics
of biological neural networks. Our results show that NeuronRank based structural
features can successfully predict average firing rates in the network, as well as the
firing rate of output neurons reflecting the network population activity. They also
indicate that link mining is a promising technique for discovering structure-activity
relations in neural information processing.

8.1 Introduction

Important functions of our brain are mediated by the operation of complex neuronal
networks. The relation between structure and function of the various types
of networks has been subject of many theories and intense computational modeling.
Fundamental questions, however, remain unanswered: How important is the
structure of a network for its function? Are certain structural features essential for
a particular function? Can one and the same structure support different functions?
Can different structures support the same function? How does the repeated usage
of a network change its structure and its function, respectively? How does the in-
teraction between networks determine the function of the whole system? These and
other, related questions are of central interest in the neurosciences.

8.1.1 Link mining and networked data

Neural networks in the brain have, at the structural level, the same format as social
networks, food webs, citation networks, the Internet, or networks of biochemical
reactions: they can be represented as large graphs, linking many interacting elements
to each other. The best studied example of such networks is probably the world
wide web. Web pages and hyperlinks between them can be considered as vertices
and directed edges, respectively, of a huge graph. Similarly, collections of scientific
publications exhibit network characteristics, as publications cite each other. Social
networks, or the transmission of epidemic diseases are further examples for networks
with nodes representing human beings.

Link mining [Getoor 2003] [Getoor 2005] is a recent direction within the field of
data mining [Witten 2005] that aims at discovering knowledge and models from ‘net-
worked’ data, which is essentially empirical data about a particular network structure
and its dynamics. Recent contributions include Kleinberg’s [Kleinberg 1999]
Hubs & Authorities algorithm, which is able to detect authoritative sources of infor-
mation on the web by exploiting its link structure and Page and Brin’s [Page 1998]
PageRank algorithm underlying the Google search engine, which successfully pre-
dicts the relevance of a web page to the user and ranks the page. The interest for
rating and ranking has also extended to classification of the linked entities. From
Chakrabarti’s et al. [Chakrabarti 1998] simple iterative relaxation labeling algo-
rithm to Taskar’s et al. [Taskar 2002] complex Relational Markov Networks, many
approaches used link information successfully for classification of web pages, research
8.1. Introduction

articles, and similar entities. Domingos' [Domingos 2005] viral marketing method assigned network values to human beings in a social network in order to optimize marketing costs by focusing the marketing on the influential customers. These are just a few examples of the tasks studied within the rapidly developing area of link mining.

8.1.2 Neuronal networks of the mammalian cortex

The cortex of mammals (‘gray matter’) is the locus of many brain functions related to sensory perception, voluntary movement control and ‘higher’ cognitive processes. Despite this broad spectrum of different functions, the structure of cortex is roughly the same everywhere, both across modalities and across species. It is clear that the limited information in the genome cannot represent all details of the sophisticated network layout directly, but it could encode general rules for the formation of neuronal connectivity during development and learning. In order to find these rules, neuroanatomists have characterized both the common and the distinguishing features of the various networks statistically [Braitenberg 1998]. In a nutshell, a local volume of 1 mm$^3$ cortical tissue contains roughly $10^5$ neurons, of which 80% are excitatory, the rest is inhibitory. The degree of convergence/divergence in the network is enormous: each neuron contacts about 10% of all the other cells, yielding a recurrent network of unprecedented complexity.

What kind of activity dynamics is implied by such networks? Sparsely coupled random recurrent networks of integrate-and-fire neurons have established themselves as computational models here. They have been studied both analytically and with the help of numerical simulations [Brunel 2000]. These networks may display different types of activity, depending on certain structural features, and on the characteristics of external input. Among other things, irregularity of individual spike trains and synchrony across neurons were identified as meaningful and easy-to-handle descriptors of collective dynamic states. Asynchronous-irregular (AI) states of network activity, in fact, are regarded the most interesting ones, since they have also been demonstrated in electrophysiological experiments in vivo.

More widespread cortical networks spanning distances of several millimeters, say, are not any more well characterized as a random graph. Long-range connections are exceedingly sparse as compared to local couplings. Consequently, viewed on a large scale, cortex is dominated by local connections, enhanced by a relatively small number of long-range connections. Networks with this specific characteristic are known as ‘small world’ networks. They can be highly clustered, like regular lattices, yet have small characteristic path lengths, like random graphs [Watts 1998]. Such networks were also shown to exhibit somewhat different dynamical states than neural networks of totally random connectivity [Barahona 2002].

The structure of large networks may be characterized in terms of small subgraph patterns, called network motifs. Figure 8.1 depicts all possible motifs involving three nodes. Motif counts can be thought of as statistical summary information for network connectivity. Song et al. [Song 2005] have shown that certain connectivity
motifs of pyramidal neurons in layer 5 rat visual cortex are in fact overrepresented as compared to random networks. Relations between motif statistics and activity dynamics of networks have also been investigated [Prill 2005]. The results indicate that the abundance or absence of particular network motifs may contribute to the stability of certain networks arising in biology, e.g. the regulatory transcription network of the bacterium E. coli, or the neural connectivity map of the worm C. elegans.

8.1.3 Motivation and scope of this chapter

We started to investigate structure-function relations in the cortex by systematically exploring the relation between network structure and activity dynamics in cortical network models. The analysis of Brunel [Brunel 2000] and others showed how the complex dynamics of a random-topology cortical network is determined by various structural parameters. In particular, the influence of the relative strength of the inhibitory synaptic couplings in the network and the role of external inputs was elucidated. The question how structural variations contribute to variations in activity dynamics, however, was not tackled in this work. Motivated by the finding that structural variations indeed influence the network dynamics [Barahona 2002]
[Prill 2005], we aimed at an automated discovery of structure-activity relations.

In this chapter we investigate the applicability of link mining techniques to reveal structure-activity relations in biological neural networks. In particular, we are interested in learning a function that maps structural features of neural networks to activity-related features. We introduce the NeuronRank [Gürel 2006] algorithm, which yields structural features describing the level to which neurons are functionally excitatory and/or inhibitory within a recurrent network. NeuronRank is inspired by the Hubs & Authorities algorithm, and is shown to yield good predictions of network activity.

We proceed by giving an overview of our approach in Section 8.2. In Section 8.3, we present our network model. We explain how we analyze the network activity in Section 8.4. We introduce our key contribution, the NeuronRank algorithm, in Section 8.5. We describe our structural feature extraction methodology in Section 8.6. We finally present our experimental results in Section 8.7 and our conclusions in Section 8.8.
Figure 8.3: Setup of the numerical simulations. We simulated recurrent cortical networks of 1,000 neurons. Each neuron in the network received external input in the form of an excitatory Poisson spike train with a total mean rate slightly above the threshold for sustained activity. All neurons in the network projected to a single ‘readout’ neuron, which did not receive any external inputs.

8.2 Overview of the method

Aiming at discovering structure-activity relations in cortical networks, we focus here on the following specific question: Which structural features of recurrent random networks are effective to predict the characteristics of their activity? Our approach is to employ link mining algorithms to find such features, and to use machine learning methods to achieve an effective prediction based on these features by learning the desired mappings from a set of examples. Figure 8.2 depicts a schematic overview of our approach.

Various structural features of specific networks were extracted, based on graph-theoretical descriptors and on the new NeuronRank algorithm. To assess the activity dynamics exhibited by these networks, we performed numerical simulations and measured the mean firing rates and other characteristic parameters based on spike statistics describing the activity dynamics. Several ‘examples’ consisting of a set of values for structural features and the corresponding activity features were then processed by machine learning algorithms, which generated statistical models for predicting the dynamics of unseen networks based on their specific structural features. We assessed the quality of these models by determining their predictive power.
8.3. The network models

8.3.1 Neuron model

We performed our simulations using the leaky integrate-and-fire neuron, which is regarded as a simple and moderately realistic model for spiking neurons, popular in the field of computational neuroscience [Brunel 2000]. The model describes the dynamics of the neuronal membrane potential based on the following differential equation

\[ \tau \dot{V}(t) = -V(t) + RI(t), \]

where \( \tau \) is the membrane time constant and \( I(t) \) is the current reflecting synaptic input to the neuron. The equivalent circuit describing the neuronal membrane is shown in Figure 8.4. Synaptic currents are induced by the spikes of the presynaptic neurons as follows

\[ RI(t) = \tau \sum_i J_i \sum_k \delta(t - t^k_i - D), \]

where \( \tau \) is the membrane time constant, \( J_i \) is the efficacy of the synapse with neuron \( i \), \( t^k_i \) are the spike times of neuron \( i \), \( D \) is a fixed transmission delay, and \( \delta \) is the Dirac delta-function.

Each spike in a presynaptic neuron \( i \) increases or decreases the membrane potential by \( J_i \). The synaptic efficacy \( J_i \) is either positive or negative depending on whether the synapse is excitatory or inhibitory, respectively. The capacitor is passively discharging through the leak resistor, with time constant \( \tau \). Whenever the voltage across the capacitor hits the threshold \( \theta \), it is instantly discharged to a value of \( U_{\text{reset}} \), and a spike is sent to all postsynaptic neurons.

The following parameters were used in all simulations: membrane time constant \( \tau = 20 \text{ ms} \), membrane leak resistance \( R = 80 \text{ M}\Omega \), spike threshold \( U_\theta = 20 \text{ mV} \), reset potential \( U_{\text{reset}} = 10 \text{ mV} \), refractory period 2 ms. Synaptic currents were modeled as delta-pulses, delayed by \( D = 1.5 \text{ ms} \) with respect to the inducing action potential.

Figure 8.4: Equivalent circuit for the integrate-and-fire neuron. In this model, spikes are generated by a simple threshold mechanism. The voltage between the lower and the upper nodes corresponds the membrane potential.
8.3.2 Types of networks

We considered three types of networks to explore structure-activity relations:

**Sparse random networks with identical synapses.** We created recurrent networks of \( n = 1,000 \) integrate-and-fire neurons, in accordance with basic statistical features of the neocortex regarding neuron types and synaptic connectivity [Braitenberg 1998]. Each of the \( n(n - 1) \) potential synapses was established with probability 0.1, independently of all the others. Neurons were inhibitory with probability 0.2, and excitatory otherwise. Excitatory neurons made only synapses with a positive efficacy on their axons, inhibitory neurons had only synapses with negative efficacies (Dale’s principle). The amplitude of all excitatory postsynaptic potentials was 0.1 mV, all inhibitory postsynaptic potentials had an amplitude of \(-0.6 \text{mV}\).

**Small-world networks.** We also generated small-world networks, starting from a locally connected ring graph with connectivity 0.1, and then randomly and independently rewiring all existing links with probability 0.1 [Watts 1998]. The relative abundance of excitatory and inhibitory neurons was the same as in the random network case, and Dale’s principle was obeyed. Inhibitory neurons were 6 times as efficient as excitatory neurons, as before.

**Sparse random networks with normally distributed synaptic efficacies.** We also experimented with random networks with normally distributed inhibitory and excitatory weights. We generated random networks as in the first setting, but then randomized synaptic efficacies by adding zero mean Gaussian perturbations.

All network simulations were performed using the NEST simulator ([Diesmann 2002] [Morrison 2005], http://www.nest-initiative.org).

8.4 Activity-related features

We studied dynamical activity ‘states’ in the networks, based on the spike firing characteristics of the neurons, in particular the mean firing rate and the firing rate of a readout neuron.

**Mean firing rate.** The networks were simulated for \( 1.2 \) s biological time. Spike counts and spike rates were determined and averaged over all neurons

\[
\nu_{\text{avg}} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{T} \int_{0}^{T} \sum_{k} \delta(t - t_i^k) \, dt,
\]

where \( n \) is the number of neurons in the network, \( T \) is the duration of simulation and \( t_i^k \) is the time of the \( k \)-th spike in the \( i \)-th neuron.

**Firing rate of a readout neuron.** The cortex is composed of many interacting local networks. It is, therefore, a relevant question how the activity of a local network affects other neurons or networks it is connected to. Here we considered the case of a single readout neuron that receives input from all neurons of a network (Figure 8.3). We were particularly interested in how the firing rate of the readout neuron depends on the structural variations of the network it receives input from.
The firing rate of the readout neuron was defined as
\[ \nu_{\text{out}} = \frac{1}{T} \int_0^T \sum_k \delta(t - t_k) \, dt, \]
where \( T \) is the duration of simulation and \( t_k \) is the time of the \( k \)-th spike of the readout neuron.

### 8.5 The NeuronRank Algorithm

The NeuronRank algorithm, which is introduced below, assigns a source value \( \alpha_i \) and a sink value \( \omega_i \) to each neuron \( i \), based only on structural information. The source value \( \alpha_i \) of a neuron encodes the net effect on the network induced by a spike in that neuron. As a rule, excitatory neurons will have positive source values, whereas inhibitory neurons will have negative source values. Exceptions to this rule, however, may exist. Namely, if an excitatory neuron excites many inhibitory neurons, it may attain a negative source value. On the other hand, if an inhibitory neuron inhibits many other inhibitory neurons, it may attain a positive source value.

The absolute source value of a neuron is an indicator for its total impact on network activity. The sink value \( \omega_i \), on the other hand, encodes the sensitivity of a neuron for activity somewhere else in the network. Neurons with higher sink values tend to be excited more by other neurons and therefore tend to have higher firing rates.

In a recurrent network, the source value of a neuron depends on the source values of all other neurons. In other words, the vector of all source values in a network recursively depends on itself, and the same holds for the vector of sink values. The NeuronRank algorithm, which we introduce below, finds a consistent set of source and sink values in a network. It iteratively updates the source value of a neuron according to the source values of its postsynaptic nodes. If \( A \) denotes the weighted adjacency matrix of the network
\[
A_{ij} = \begin{cases}
1 & \text{for an excitatory synapse } j \rightarrow i \\
-g & \text{for an inhibitory synapse } j \rightarrow i \\
0 & \text{otherwise},
\end{cases}
\]
where \( g > 0 \) is a number that encodes the relative impact of inhibitory couplings relative to excitatory ones. The update rule for the row vector of source values \( \alpha = (\alpha_1, \ldots, \alpha_n) \) is given by
\[ \alpha \leftarrow \alpha A \]
starting with initial values \( \alpha_i \pm 1 \) depending on whether neuron \( i \) is excitatory or inhibitory. In contrast to source values, the sink value of a neuron is updated according to the sink values of its presynaptic nodes. The update rule for the column vector of sink values \( \omega = (\omega_1, \ldots, \omega_n)^T \) is therefore given by
\[ \omega \leftarrow A \omega, \]
Table 8.1: The NeuronRank algorithm.

<table>
<thead>
<tr>
<th>Input:</th>
<th>A directed labeled (inhibitory/excitatory) recurrent network $N$, represented by a weighted adjacency matrix $A$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output:</td>
<td>Source ($\alpha$) and sink ($\omega$) values of all nodes in $N$</td>
</tr>
</tbody>
</table>

for each node $i$ in $N$

$$\omega_i \leftarrow 1$$

if $i$ is excitatory

$$\alpha_i \leftarrow 1$$

else if $i$ is inhibitory

$$\alpha_i \leftarrow -1$$
endif
endif
endfor
repeat

$$\alpha \leftarrow \alpha A$$

$$\omega \leftarrow A \omega$$

normalize $\alpha$ and $\omega$ such that $\sum_i \alpha_i^2 = 1$ and $\sum_i \omega_i^2 = 1$
until convergence
return $\alpha$ and $\omega$

starting with initial values $\omega_i = 1$ for all neurons. In each step of the iteration both $\alpha$ and $\omega$ are normalized to unit length, and the iteration stops upon convergence. The detailed algorithm is depicted in Table 8.1.

8.6 Structural features

8.6.1 NeuronRank features

Upon convergence of the NeuronRank algorithm, statistical summary information about the source and sink values in a network is passed on to data mining algorithms. We considered in particular mean and variance of the source values, as well as mean and variance of the sink values of all neurons. In addition, sum, mean and variance were computed separately for excitatory and inhibitory neurons, respectively. This yielded a set of total 16 structural features.

8.6.2 Network motifs

Network motifs [Milo 2002] are subnetworks involving a small number of neurons. Formally, a network motif of $n$-th order is an isomorphic set of unlabeled directed graphs $G = (V, E)$, where $V = \{v_1, v_2, ...v_n\}$ is the set of nodes and $E$ is the set of...
edges. Consider a network with $N$ nodes. Each of the $\binom{N}{n}$ different subnetworks with $n$ nodes is then isomorphic to exactly one order $n$ motif. The motif count $k_G$ is the number of subnetworks that are isomorphic to one particular $G$. Examples of network motifs are shown in figure 8.1. Network motifs can be defined and counted in labeled networks as well, e.g. in neural networks where neurons have different types. In this case, label matching is part of equivalence checking.

We implemented an algorithm, which counts second order and third order motifs in a network. Motif counting was performed both with and without considering neuron types. Note that there are 16 third-order network motifs without considering the types, and 93 third-order network motifs considering the neuron types.

8.6.3 Cluster coefficient

We also computed the average cluster coefficient [Watts 1998] of each network and used it as a feature for machine learning algorithms. The cluster coefficient was originally defined on undirected graphs, here we use the following straightforward extension to directed graphs

$$C = \frac{1}{N} \sum_{n} \frac{E_n}{k_n(k_n-1)},$$

where $k_n$ is the number of neighbors of neuron $n$ (either presynaptic or postsynaptic), and $E_n$ is the total count of synapses among the neighbors. The latter number is bounded by $k_n(k_n-1)$. The coefficient $\frac{E_n}{k_n(k_n-1)}$, therefore, yields a measure of clustering from the perspective of neuron $n$. The average $C$ of these coefficients for all neurons yields a measure for the cliquishness of the network [Watts 1998].

8.7 Experimental results

In order to tackle the question whether structural features, including those obtained from the NeuronRank algorithm, are good predictors for the network activity, we applied well-known machine learning algorithms implemented in the WEKA workbench for data mining [Witten 2005] to the four structure-activity relation prediction tasks sketched below. The machine learning algorithms employed were J48, a decision tree learner, the K2 algorithm [Cooper 1992] for learning Bayesian Networks, and Support Vector Machines [Cristianini 2000b] for predicting the class variable of the networks (e.g. ‘high firing rate’ or ‘low firing rate’). J48 is the java version of C4.5 decision tree learning algorithm [Quinlan 1993], which induces a tree in a top-down manner using information gain as its heuristic. C4.5 supports attributes with numerical values and performs tree post-pruning. K2 is a hill climbing based learning algorithm for Bayesian networks, which can deal with missing data. K2 uses a special ordering of the attributes to guide its search when adding arcs to the Bayesian network. In all experiments listed below, we report on the predictive accuracy of these algorithms using 10-fold cross-validation.
Predicting the average firing rate in sparse random networks with identical synapses. We generated 330 random networks and performed numerical simulations of their activity dynamics. For each network, we measured the firing rate averaged across all neurons. Firing rates above the median were labeled as ‘high’, below the median as ‘low’. The task then was to predict the firing rate correctly (‘high’ vs. ‘low’), based on the features extracted by motif counting (cc: clustering coefficient, inh: inhibitory neuron count, 3m: third order motifs, 2ie: second order motifs with signs, 3ie: third order motifs with signs) and by the NeuronRank algorithm (mean, variance and sum of source and sink values). Note that running NeuronRank took 25.2 seconds per network on average, whereas counting third order type-specific motifs took 255.2 seconds per network on average (Pentium-4 3.2 GHz processor, SuSE Linux 10.0 operating system). The results are shown in Table 8.2.

Predicting the firing rate of a readout neuron. In the numerical simulations, a readout neuron was added to each of the networks described above. This neuron received input from all neurons in the network, but no external input (see also Figure 8.3). We considered the same structural features as in the previous setting (ignoring the readout neuron) and trained the machine learning algorithms on examples to predict the firing rate of the readout neuron as ‘low’ or ‘high’ on unseen networks. The results are shown in Table 8.2.

Predicting the firing rate in small-world networks. We generated small world networks as explained in Section 8.3 and simulated their activity with low-frequency Poisson input. We used the same structural features and discretized the firing rate into ‘high’ and ‘low’, as described above. Prediction accuracies are shown in Table 8.3.

Predicting the average firing rate in sparse random networks with continuous synaptic efficacies. The same features, discretization, and input were applied as in the first experiment. However, normally distributed synaptic efficacies were imposed. The task was again to predict the firing rate as either ‘high’ or ‘low’. Table 8.4 shows the outcome of this experiment.

The results of our experiments clearly demonstrate the predictive power of the NeuronRank features. For all experiments, NeuronRank features contributed significantly to the accuracy of the prediction, so we can regard them as excellent indicators of network activity. NeuronRank features, when used together with inhibitory neuron count (inh), was the best of all feature combinations for activity prediction. 3-rd order type-specific motif counts (3ie), also used together with inhibitory neuron count (inh), had the closest prediction accuracy to the NeuronRank features in general. This is not surprising since 3ie feature set contains the most detailed and the most complete summary of the network structure among motif counting based features. Some machine learning algorithms, such as decision trees (J48) and Bayesian networks (K2), are more sensitive to large numbers of features than support vector machines. This explains why the performance gain for SVMs when working with 3-rd order type specific motifs was typically larger.

Note that NeuronRank based best features (inh +source values+sink values) outperformed motif counting based best features (inh +3ie) in prediction. The
Table 8.2: Accuracy of prediction in sparse random networks using three well known machine learning algorithms: K2 Bayesian Network algorithm, J48 decision tree learner, and Support Vector Machines (SVM) using first and second order polynomial kernels. Note that the presented SVM results are the better ones from both types of kernels. cc: clustering coefficient, inh: inhibitory neuron count in the network, 2ie/3ie: second/third order motifs respecting node types, 3m: third order motifs ignoring node types.

<table>
<thead>
<tr>
<th>Features</th>
<th>Average Firing Rate</th>
<th>Readout Firing Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BN-K2</td>
<td>J48</td>
</tr>
<tr>
<td>cc</td>
<td>48.5%</td>
<td>48.5%</td>
</tr>
<tr>
<td>inh</td>
<td>87.0%</td>
<td>87.0%</td>
</tr>
<tr>
<td>inh + cc</td>
<td>87.0%</td>
<td>86.7%</td>
</tr>
<tr>
<td>inh + 2ie</td>
<td>89.7%</td>
<td>91.5%</td>
</tr>
<tr>
<td>inh + 3m</td>
<td>87.0%</td>
<td>85.8%</td>
</tr>
<tr>
<td>inh + 3ie</td>
<td>86.7%</td>
<td>90.1%</td>
</tr>
<tr>
<td>inh + source values</td>
<td>92.7%</td>
<td>94.8%</td>
</tr>
<tr>
<td>inh + sink values</td>
<td>93.0%</td>
<td>93.0%</td>
</tr>
<tr>
<td>inh + source + sink values</td>
<td>92.1%</td>
<td>93.0%</td>
</tr>
<tr>
<td>source values</td>
<td>92.4%</td>
<td>93.0%</td>
</tr>
<tr>
<td>sink values</td>
<td>90.9%</td>
<td>92.4%</td>
</tr>
<tr>
<td>source + sink values</td>
<td>92.1%</td>
<td>93.3%</td>
</tr>
</tbody>
</table>
Table 8.3: Accuracy of prediction in small-world networks. Abbreviations denote the same features and algorithms as in Table 8.2.

<table>
<thead>
<tr>
<th>Features</th>
<th>Average Firing Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BN-K2</td>
</tr>
<tr>
<td>cc</td>
<td>50.0%</td>
</tr>
<tr>
<td>inh</td>
<td>84.8%</td>
</tr>
<tr>
<td>inh + cc</td>
<td>84.8%</td>
</tr>
<tr>
<td>inh + 2ie</td>
<td>87.0%</td>
</tr>
<tr>
<td>inh + 3m</td>
<td>84.8%</td>
</tr>
<tr>
<td>inh + 3ie</td>
<td>86.0%</td>
</tr>
<tr>
<td>inh + source values</td>
<td>84.5%</td>
</tr>
<tr>
<td>inh + sink values</td>
<td>80.5%</td>
</tr>
<tr>
<td>inh + source + sink values</td>
<td>83.3%</td>
</tr>
<tr>
<td>source values</td>
<td>78.5%</td>
</tr>
<tr>
<td>sink values</td>
<td>65.8%</td>
</tr>
<tr>
<td>source + sink values</td>
<td>83.3%</td>
</tr>
</tbody>
</table>
### 8.7. Experimental results

Table 8.4: Accuracy of prediction in networks with normally distributed synaptic efficacies. Abbreviations denote the same features and algorithms as in Table 8.2.

<table>
<thead>
<tr>
<th>Features</th>
<th>Average Firing Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BN-K2</td>
</tr>
<tr>
<td>cc</td>
<td>48.5%</td>
</tr>
<tr>
<td>inh</td>
<td>86.4%</td>
</tr>
<tr>
<td>inh + cc</td>
<td>86.4%</td>
</tr>
<tr>
<td>inh + 2ie</td>
<td>87.6%</td>
</tr>
<tr>
<td>inh + 3m</td>
<td>86.4%</td>
</tr>
<tr>
<td>inh + 3ie</td>
<td>87.0%</td>
</tr>
<tr>
<td>inh + source values</td>
<td>88.5%</td>
</tr>
<tr>
<td>inh + sink values</td>
<td>89.7%</td>
</tr>
<tr>
<td>inh + source + sink values</td>
<td>89.4%</td>
</tr>
<tr>
<td>source values</td>
<td>86.7%</td>
</tr>
<tr>
<td>sink values</td>
<td>87.3%</td>
</tr>
<tr>
<td>source + sink values</td>
<td>87.6%</td>
</tr>
</tbody>
</table>
Table 8.5: Comparison of the best NeuronRank based feature set (inh+source values +sink values) to the best motif counting based feature set (inh+3ie) according to sign test. NeuronRank features were significantly better at binary prediction task.

<table>
<thead>
<tr>
<th># NeuronRank better</th>
<th># 3ie better</th>
<th>p value</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>1</td>
<td>0.0032</td>
</tr>
</tbody>
</table>

difference was statistically significant according to sign test applied to those two feature sets with a p-value of 0.0032 (see Table 8.5).

8.8 Conclusions

We showed that it is possible to model certain aspects of the activity dynamics in random cortical networks by employing data mining and machine learning techniques. Furthermore, we demonstrated that NeuronRank, which is related to the Hubs & Authorities and PageRank, can successfully extract structural features that are relevant for predicting certain aspects of network activity. This indicates that learning algorithms can relate the neural structure (which is in the format of networked data) to certain features of neural activity. Structural features can be successfully extracted by link mining methods. Our results indicate that employed link mining methods in the discovery of structure-activity and structure-function relations is useful. Information extraction tools, which are designed for networked data, can help neuroscientists to gain insight to computations performed by sophisticated neural structures. Since recent simulation results show how specific neural structures result in precise firing patterns in cortical neural networks [Izhikevich 2006, Diesmann 1999], employing link mining algorithms would be interesting and promising to discover structural features that predict further functional aspects such as precise spike timing. Building on our experiences with simulated activity data, we are currently adapting our algorithms to discover structure-activity relations in biological neuronal networks, like cell cultures grown on multi-electrode arrays [Marom 2002].
In this thesis, we provided some examples of adaptive computing tools for functional modeling of biological neural networks. We utilized simulated networks and dissociated cortical cultures as reference biological networks. Echo State Networks was used as the modeling tool. We had listed our findings and contributions in the introductory chapter. Here, we are going to discuss them in more details.

- We show that point process framework is appropriate for predicting the spike activity in neuronal cultures. We introduced a simple point process based online learning algorithm to predict the density of the spiking activity in dissociated cortical cultures (chapter 4).

- We show that it is possible to deliver predictive functional models of biological neural networks (chapters 6 and 7). Our findings indicate that reservoir computing is a potential candidate for modeling neural activity including neural encoding and decoding. ESNs with leaky integrator neurons can be sensitive to different activity measures (e.g. spike rates, correlations etc.) and thereby can be a generic tool for different subproblems in functional modeling, including neural decoding. As reservoir computing tools are dynamical systems, they can also be used as simulation tools for modeling network dynamics from neural input, i.e. they can be applied for neural encoding. For instance Jarvis et al. showed that, with appropriate reservoir structuring, ESNs could reflect the bursting behavior of the dissociated cortical cultures. Using ESNs as the tool, Jarvis et al. identified the structural criteria for bursts to occur [Jarvis 2009].

- We adapted the ESN readout learning algorithms for point process data based on point process log-likelihood (chapter 6). Traditionally, ESN algorithms are designed for continuous valued signals. For data of point events in time, more suitable algorithms have been introduced (see chapter 3). We adapted the ESN readout learning for this data. Note that the adaptation is quite trivial, if it is only on readout parameters, since the problem then is equivalent to learning the parameters of a Generalized Linear Model (GLM) [Paninski 2004, McCullagh 1989]. For reservoir adaptation, however, the task is not trivial due to recurrent connectivity.

- We showed how to efficiently adapt the reservoir parameters based on the propagation of the point process log-likelihood into the reservoir. Note that in
a recurrent network, back-propagation based algorithms are computationally expensive. For a sparse network and point process data, however, we show that gradients can be well approximated with one-step propagation (chapter 6). The proposed algorithm was used to adapt connectivity and time constants of the reservoir neurons. The experimental results showed that reservoir adaptation can significantly improve the ESN performance over readout-only training. The adaptive reservoirs significantly outperformed the readout-only-trained reservoirs for most of the tasks. Here, we used one-step log-likelihood propagation into the reservoir, which was an approximation to the full gradient. The full gradient computation would be the point process version of Real Time Recurrent Learning (RTRL) [Williams 1989], which is a very expensive algorithm for our data and reservoir sizes. For sparse reservoirs and point process data, we showed that one-step propagation approximated the gradients pretty well.

• We proposed feed-forward reservoirs and demonstrate that for some tasks, sparse feed-forward reservoirs are better adaptable by one-step log-likelihood propagation than recurrent ones. We also analyzed the underlying reasons for that and experimentally showed that gradients are better approximated by one-step log-likelihood propagation in feed-forward reservoirs (chapter 6). In our comparison of feed-forward reservoirs to recurrent ones, we can think of no obvious starting point disadvantage for recurrent reservoirs. We experimentally investigated reasons why feed-forward adaptation might be advantageous. When we analyzed the similarity of the gradients approximated by one-step propagation with those obtained by full gradient computation, we saw that, in feed-forward reservoirs, the one-step-gradients were closer to the full gradients. We argue that this is caused by the better structuring of the reservoir parameters. In feed-forward reservoirs, the memory and the connectivity parameters are not coupled anymore. There, memory is due to the time constants of the individual leaky neurons. In recurrent reservoirs, on the other hand, the memory is also due to recurrent network connectivity. This might cause a small connectivity change in the recurrent adaptation to have a more dramatic effect on the reservoir memory than in case of feed-forward adaptation. Recently, there has been some research on the information properties of feed-forward neural architectures. Goldman and Ganguli et al. [Ganguli 2008, Goldman 2009, Murphy 2009] showed that stable fading memory can be realized by feed-forward or functionally feed-forward networks and that feed-forward networks have desirable properties in terms of trade-off between noise amplification and memory capacity. The better performance of feed-forward networks in our experiments are in accordance with their findings.

• We showed that models for dynamic attractor states of dissociated cortical cultures can be developed by reservoir computing methods. The global firing trajectory could be related to a one-time-step cue signal (chapter 7). In response to the one-time-step input, the ESN generated a temporary oscillation
that reflects the attractor state of the reference biological network. Extension of modeling neural systems to state trajectories is a significant feature for the analysis of neural data and for neurotechnological applications. For instance, prediction of temporal patterns might provide better interception with biological neural networks, such as epileptic seizures or burst elimination in cultures. This thesis demonstrated that reservoir dynamics reveals the underlying structure in the recorded data. Different types of attractors were encoded in different trajectories of pairwise reservoir unit phase plots. This was due to higher dimensionality of the reservoir. We argue that reservoir dynamics might reflect possible sources and structures in the observed neural activity, which cannot be distinguished directly from recordings. Another future improvement of our work would be broadening models of temporal pattern attractors to those of spatio-temporal ones. For this purpose, more parameter engineering and computing power have to be invested in this direction.

- Extending our main focus on functional modeling of biological neural networks, we proposed an adaptive computing method for mining the structure of biological neural networks. We proposed the NeuronRank algorithm, for extracting structural features from a biological neural network. NeuronRank assigns source and sink values to every neuron in the network in an iterative scheme. The proposed algorithm is neural analogous of the link mining algorithms PageRank [Page 1998] and Hubs & Authorities [Kleinberg 1999]. Using NeuronRank, we showed that adaptive computing methods are not restricted to neural activity but they can also extend to neural connectivity (chapter 8). Using network simulations, we showed that NeuronRank extract features that are predictive for activity parameters of the network.

We took a reservoir computing approach to the functional modeling of BNNs. In the light of our findings, we believe that reservoir computing can be applied to a wide range of problems in computational neuroscience, especially to neural encoding and decoding. Brain Machine Interfaces (BMI) are one of the areas, where reservoir computing could be potentially successful. Beside new application areas, improvements in the core reservoir computing area can extend the use of this framework in computational neuroscience. Here, we can identify two major directions in the reservoir computing area, which can enhance our understanding of and interaction with BNNs including brains.

**Improvements in biological plausibility** One of the drawbacks of LSMs in comparison to ESNs is the inapplicability of the gradient based algorithms to reservoirs, as the activation functions of the individual LSM units are usually not differentiable (IAF neurons). This drawback, however, is a consequence of more realistic neuron models in LSMs compared to ESNs. More realistic neuronal modeling also allows for biologically more plausible learning rules in the LSM framework. There has recently been some research effort on
combining biologically realistic reward modulated spike timing based learning rules and random networks of spiking neural networks, i.e. spiking reservoirs [Izhikevich 2007]. This new combination has been shown to be successful in explaining biological phenomena such as classical and instrumental conditioning, reinforcement learning and response shift. In another work, the same learning rule also explained cortical reorganization in a monkey brain-control task [Legenstein 2010].

**Improvements in reservoir adaptation methods** Development of learning rules which improve the performance of random reservoirs enhances the applicability of reservoir computing to biological problems. For instance, Sussillo and Abbot [Sussillo 2009] have recently developed the FORCE learning algorithm, which adapts the reservoir to generate coherent movement patterns from chaotic spontaneous activity. Their results reproduce the data of the premovement activity in the motor cortex.
Implementation details of the Echo State Networks in chapter 6

For all the experimental settings, we generated recurrent sparse random reservoirs. Regardless of the reservoir size, each reservoir unit was connected to 10 other units. The initial connection strengths were drawn from a uniform distribution over the interval $[-0.5, 0.5]$. The entries of the input weight matrix $W^{in}$ were drawn from a uniform distribution over $[-1, 1]$. We normalized the connectivity so that the maximum absolute eigenvalue, i.e. the spectral radius, was 1, with which the ESNs were experimentally detected to work well on all experiments. We additionally generated the feed-forward form of the each recurrent reservoir as described in figure 6.5. After changing from recurrent to feed-forward architecture, we did not manually modify the connectivity parameters. The leakage time constants, $\tau_j = 1/\alpha_j$, were initialized according to

$$\alpha_j = 1/(1 + \exp(\alpha'_j)),$$

where $\alpha'_j$ were uniformly drawn from $[-1.5, 1.5]$.

Note that the initial value of the spectral radius and the initial range for time constants were manually arranged to work well with recurrent fixed reservoirs. Adaptive reservoirs used this fixed settings as starting parameters.

For fixed reservoirs, we used 80 epochs of readout-only training. Initially, we set $\eta_{out} = 0.7$ and gradually decreased the output learning rate with

$$\eta_{out} \leftarrow \frac{\eta_{out}}{E + 1},$$

where $E$ was the epoch number. This was done for fast learning and convergence of the algorithm.

For adaptive reservoirs, we used 20 epochs for reservoir and readout training and 60 epochs for readout-only training. The readout-only learning employed the same time dependent $\eta_{out}$ as the fixed reservoirs. For reservoir training, we set the initial values $\eta_{out} = \eta_{res} = \eta_{a} = \eta = 0.2$ and decreased it with

$$\eta \leftarrow 0.5 \eta,$$

when the increase in the training log-likelihood dropped below 0.0003 per output channel and time step.

The parameter $A$ in $f^{out}(\xi) = \exp(A\xi)$ was set to 0.2, which was manually found to work well for the selected learning rates.
Concavity of the log-likelihood with respect to readout parameters for ESN readout training in chapter 6

The second order partial derivative of the instantaneous log-likelihood with respect to output parameters is computed as

$$\frac{\partial^2 \ell}{\partial W_{ij} \partial W_{ik}} = -A^2 x_j^n x_k^n \lambda_i^n,$$

resulting in a Hessian,

$$H_{\ell_i} = -A^2 \lambda_i^n [x^n; u^n] [x^n; u^n]^T,$$

where $[x^n; u^n]^T$ is the transpose of the column vector $[x^n; u^n]$. $H_{\ell_i}$ is obviously a non-positive definite matrix, since for any non-zero row vector $b$

$$b \ H_{\ell_i} \ b^T = -(A^2 \lambda_i^n) \ b [x^n; u^n] [x^n; u^n]^T \ b^T$$

$$= -(A^2 \lambda_i^n) \ (b [x^n; u^n])^2$$

$$\leq 0.$$

Consequently, the Hessian of the total log-likelihood is also non-positive definite as $H_{\mathcal{L}} = \sum_i \sum_n H_{\ell_i}$. Therefore, we conclude that under fixed reservoir parameters, the log-likelihood is a concave function with respect to output parameters. This indicates that output parameters will not get stuck in the local maxima for a fixed reservoir. For adaptive reservoirs, the concavity holds for the readout-only learning phase of the training. An overview of the concavity of the point process log-likelihood for generalized linear models and its extension to different conditional intensity functions is given in [Paninski 2004].
Bibliography


[Cristianini 2000a] N Cristianini and J Shawe-Taylor. ... *Introduction to Support Vector Machines: and other kernel-based learning* ... books.google.com, Jan 2000. 5


neuron groups: winnerless competition. Physical review letters, vol. 87, no. 6, August 2001. 8, 82


