DISSERTATION

to obtain the title of

Doctor of Technical Sciences

of

Defended by

Stefan KLAMPFL

Computation and Learning in Biological Networks of Neurons: Theoretical Analysis, Computer Simulations, and Analysis of Experimental Data

Thesis Advisor:

O.Univ.-Prof. Dipl.-Ing. Dr.rer.nat. Wolfgang MAASS

defended on October 7, 2011

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, September 2011 .....

(signature)

Abstract

Computations in the brain differ fundamentally from those in traditional digital computers. Most notably, the brain is organized in a massively parallel manner and has the ability to learn. The Liquid State Machine has emerged as a powerful model that provides a framework for explaining computation and learning in biological networks of neurons: Recurrent networks of spiking neurons can serve as generic preprocessing units that allow simple, typically linear readout neurons of these networks to be adapted for complex computational tasks. This thesis makes important contributions to this framework in two ways. It investigates a number of unsupervised learning algorithms, which are potential candidates for such readout mechanisms, and provides novel experimental evidence for this computing model using data from the primary auditory cortex of awake ferrets.

First, it is shown how two unsupervised learning mechanisms, information bottleneck optimization and independent component analysis, can in principle be implemented using biologically realistic neuron models by deriving suitable learning rules from these abstract information theoretic principles. The resulting learning rules are analyzed theoretically and tested in a number of computer simulations.

Second, slow feature analysis is investigated as another unsupervised learning principle. A theoretical analysis shows that under some conditions on the statistics of the input time series it is able to achieve the classification capability of a well-known supervised learning method, Fisher's linear discriminant. Furthermore, readouts of a computer model of a cortical microcircuit trained with this method are able to learn to detect repeating firing patterns within a stream of spike trains with the same firing statistics and to discriminate between the network responses to different stimuli in a completely unsupervised manner.

Finally, biological data from neurons in the primary auditory cortex of ferrets are analyzed using state-of-the-art methods from machine learning and information theory. It is shown that sequentially arriving stimulus information is integrated over time and superimposed in a non-linear way into the neural responses at one point in time. This provides thus experimental evidence for the liquid computing model, for the first time using data from awake animals.

Zusammenfassung

Berechnungen im Gehirn unterscheiden sich fundamental von denen in traditionellen digitalen Computern. Vor allem ist das Gehirn auf massiv parallele Weise organisiert und hat die Fähigkeit zu lernen. Die "Liquid State Machine" ist als ein mächtiges Modell entwickelt worden, das einen Rahmen zur Verfügung stellt, um Berechnungen und Lernen in biologischen Netzwerken von Neuronen zu beschreiben: Rekurrente Netzwerke von spikenden Neuronen können als generische Vorverarbeitungseinheiten dienen, die es einfachen, typischerweise linearen "Readout"-Neuronen dieser Netzwerke erlauben, sich für komplexe Berechnungsaufgaben zu adaptieren. Diese Dissertation liefert in zweierlei Weise wichtige Beiträge zu diesem Framework. Einerseits untersucht sie eine Anzahl von unüberwachten Lernalgorithmen, die potentielle Kandidaten für solche Readout-Mechanismen sind, andererseits liefert sie neue experimentelle Belege für dieses Berechnungsmodell basierend auf Daten des primären auditorischen Kortex von wachen Frettchen.

Zunächst wird gezeigt, wie zwei unüberwachte Lernmechanismen, Information Bottleneck Optimierung und Independent Component Analysis, im Prinzip mit biologisch realistischen Neuronenmodellen implementiert werden können, indem geeignete Lernregeln von diesen abstrakten informationstheoretischen Prinzipien hergeleitet werden. Die resultierenden Lernregeln werden theoretisch analysiert und in einer Reihe von Computersimulationen getestet.

Darüber hinaus wird Slow Feature Analysis als ein weiteres unüberwachtes Lernprinzip untersucht. Eine theoretische Analyse zeigt, dass dieses unter einigen Bedingungen an die Statistik der Inputzeitserie die Klassifikationsfähigkeit einer bekannten überwachten Lernmethode erlangen kann, Fishers linearen Diskriminante. Außerdem sind Readouts eines Computermodells eines kortikalen Mikroschaltkreises, die mit dieser Methode trainiert worden sind, in der Lage, sich wiederholende Feuermuster innerhalb eines Inputstroms mit derselben Feuerstatistik zu detektieren, sowie zwischen den Netzwerkantworten zu verschiedenen Stimuli in komplett unüberwachter Weise zu unterscheiden.

Abschließend werden biologische Daten von Neuronen aus dem primären auditorischen Kortex von Frettchen mit modernsten Methoden des Maschinellen Lernens und der Informationstheorie analysiert. Es wird gezeigt, dass sequentiell ankommende Stimulusinformation in die neuronale Antwort zu einem bestimmten Zeitpunkt über die Zeit integriert und auf nichtlineare Weise kombiniert wird. Dies liefert daher einen experimentellen Beleg für das "Liquid Computing" Modell, zum ersten Mal aufgrund von Daten von wachen Tieren.

Acknowledgements

Above all, I would like to thank my advisor Wolfgang Maass for his inspiring research ideas, for his continuous support during my studies, and for giving me the opportunity to participate in top-level research.

Many thanks go to Robert Legenstein, Stephen V. David, Pingbo Yin, and Shihab A. Shamma for enjoyable and fruitful collaborations as well as their guidance. I am very grateful for Laurenz Wiskott, who gave me the possibility to visit his exciting research team and who provided particularly helpful comments on Chapter 3 of this thesis. Additionally, I would like to thank Gordon Pipa for being the second referee of this thesis.

Special thanks go to all of my current and former colleagues during my time at the Institute for Theoretical Computer Science, especially to Stefan Häusler, Klaus Schuch, Gregor Hörzer, Michael Pfeiffer, Roland Unterberger, Martin Ebner, Oliver Friedl and Daniela Potzinger, all of who made my life so enjoyable and who became friends also outside of research. I also want to acknowledge that this thesis would not have been possible without the financial support from Graz University of Technology, the Austrian Science Fund FWF, and various research programs of the European Union, including FACETS, SECO, and ORGANIC.

Last but not least, I want to express my deepest gratitude to my family and friends, in particular to my parents, Gerhard and Hermine, for their endless patience, encouragement, and support they have provided to me in all possible ways throughout my whole life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One major challenge in the field of neuroscience is to understand how computations are carried out in the brain. One possible approach to this question is to view the computational organization of the brain from the perspective of a digital computer. On an abstract level there are indeed some analogies: Much like in a computer the brain receives input (through sensory organs), processes and stores information, and produces an output (e.g., by activating certain muscles). On a closer look, however, there are major differences between both systems. Perhaps the most apparent difference is the massively parallel organization of the brain: billions of basic processing units ($\sim 10^{11}$ *neurons*) are organized in recurrently connected networks (through $\sim 10^{15}$ *synapses*) without an obvious uniform synchronization mechanism. Furthermore, these neurons are not identical, but vary substantially in their anatomical and physiological properties (Markram et al., 2004). Such heterogenity of components is typically not present in digital computers. The same applies for the connections between these neurons, the synapses, whose strength is not fixed but is plastic and varies depending on the activity of those neurons participating in the connection (Hebb, 1949). Moreover, synapses are often not sufficiently characterized by this strength as a single scalar value, rather they undergo complex inherent temporal dynamics on its own (Markram et al., 1998). This continuous adaptation and permanent retuning is widely considered the basis for learning in biological organisms. Thus, computation and learning in neural systems are apparently strongly interconnected.

The classical mathematical model that describes computation in common digital computers is the *Turing machine* (Sipser, 1996). During a computation the Turing machine undergoes a certain sequence of states while modifying the content of a (potentially infinite) tape. At the beginning the input is presented on this tape, and the content when a halting-state is reached is considered the output of the computation. However, typical operations in the brain are largely different. First, inputs are not presented at designated time steps, but continuously arrive all the time from different pathways, e.g., from the visual system, from the muscles, also from memory, etc. Usually many of these input components have to be integrated into a computation, and new computations start before other ones are finished. Furthermore, biological organisms often cannot wait for the results of computations, but have to perform actions within a fixed time interval, e.g., to avoid stumbling over an obstacle or to react to an immediate threat.

Having this in mind it becomes apparent that classical computational models, such as the Turing machine, are not adequate for describing and understanding brain-style computations. One computational model that has been developed to describe exactly these type of computations is the *Liquid State Machine* (LSM)

(Maass et al., 2002, 2004b; see Maass, 2007, 2010, for reviews). Rather than strings (or numbers), the inputs and outputs of an LSM are functions of time: input streams $u(t)$ are mapped onto output streams $y(t)$. Each LSM consists of two parts: (i) a dynamical system (the "liquid"; which could be a sufficiently large and diverse recurrent network of biological neurons) which integrates previously arrived input $u(s)$, $s \leq t$, into a high-dimensional *liquid state* $\mathbf{x}(t)$, and (ii) a static, memoryless *readout* that instantaneously transforms the liquid state into the output $y(t) = f(\mathbf{x}(t))$. It can be shown that under some mild conditions on the liquid the readout function f can be chosen to approximate any Volterra series (Maass and Markram, 2004), and if one additionally allows feedback from the readout into the liquid, a large class of dynamical systems can be emulated (Maass et al., 2007).

From the perspective of biological networks of neurons this computing model is attractive because it allows to view cortical circuits as generic preprocessing components that integrate incoming information over time into a liquid state at time t, thereby making this information available to downstream neurons that can then simultaneously and instantaneously "read out" different pieces of information about previously arrived input. A cortical circuit can support such readout neurons by (i) providing analog fading memory to accumulate information over time in the liquid state, and (ii) a nonlinear projection into high-dimensional space to ease the extraction of information by readouts (i.e., to serve as a *kernel* in the terminology of machine learning). With such a preprocessing, substantial computational power can be gained even by simple linear readouts, which are a reasonable approximation to biological neurons since they compute a weighted sum of their presynaptic spike trains and produce an output once this sum exceeds a certain threshold. There exists experimental evidence that circuits in the brain of living animals exhibit both properties, e.g., neurons in the primary visual cortex of anesthetized cats have been shown to nonlinearly combine information about previous stimuli and to maintain this information for several 100ms (Nikolic et al., 2009).

Another major advantage of the LSM over traditional computational models is that it is an *adaptive* computing system. Readout mechanisms can be optimized for a given computational task, for example by adjusting the weights of input synapses if the readout is a model of a biological neuron. There are several ways how such readouts can be trained. The most promising approach from the perspective of machine learning is to use *supervised learning*, where the readout function is inferred from a training set consisting of a number of training samples paired with corresponding target values. In case of an LSM the training set typically consists of snapshots of the activity of a recurrent network of neurons (i.e., liquid states) at particular points in time in response to some input patterns. The readout should then predict these labels for unseen patterns as accurately as possible. This has been successfully applied to a number of applications, including speech recognition (Maass et al., 2002, 2004a; Legenstein et al., 2005). However, from the perspective of biological neural systems this method is not very realistic, since the existence of a supervisor that tells the brain how an external stimulus should be classified is highly questionable. There exist biologically realistic learning rules that have been shown to achieve a reasonable performance with substantially less amount of supervision (Legenstein et al., 2008). Of particular interest are completely *unsupervised*

learning mechanisms, which usually perform an optimization with respect to some statistics of the input (or liquid states, in the case of readouts). Such unsupervised learning rules are thought to play an essential role in the processing of information in the brain, but for many of these it has remained unclear how such mechanisms could actually be implemented in networks of biologically realistic neurons.

This thesis makes important contributions to the field of computation and learning in biological networks of neurons in two ways. First, it is shown how a number of unsupervised learning algorithms can be implemented using biologically realistic neuron models, and for one algorithm its capability as a possible readout mechanism of cortical microcircuits is analyzed. Second, new experimental evidence for the liquid computing model is reported using data from the primary auditory cortex of awake ferrets.

1.1 Organization of the Thesis

This thesis is comprised of three chapters which are based on publications to which I significantly contributed as first author during my PhD studies.

In Chapter 2, two unsupervised learning mechanisms are investigated that are based on information theoretic principles, information bottleneck optimization and independent component analysis. The information bottleneck method tries to select a compact representation of some input data, while preserving as much relevant information as possible. A neuron that performs information bottleneck optimization extracts preferentially those components from high-dimensional input streams that are related to (i.e., have high mutual information with) a specific relevance or target signal. On the other hand, independent component analysis is a technique for decomposing complex data into statistically independent parts, thereby providing a less redundant representation. Two neurons extract independent components from its common input if the mutual information between its output spike trains is low. By deriving learning rules from these related abstract information theoretic principles, it is shown how spiking neurons can in principle perform both of these tasks. A theoretical analysis revealed that the resulting learning rules are extensions of the well-known BCM-rule, a variant of Hebbian learning that intrinsically stabilizes the ouput firing rate through a sliding threshold. In addition, the learning rules were tested in a number of computer simulations that demonstrated that the learning neurons were sensitive to information encoded both in the firing rate and in the spike timing of their inputs.

Another powerful unsupervised learning principle is temporal slowness, which is the topic of Chapter 3. This principle is based on the assumption that the slowest components of a high-dimensional signal typically encode invariances of this input. For example, the identity of an object in the visual field generally varies on a much slower time scale than the raw sensory input because temporally contiguous input samples are likely to be caused by the same object. We present a theoretical basis for this emergent discrimination capability by showing that one particular algorithm from the family of temporal slowness learning methods, slow feature analysis (SFA), is able to approximate the classification capability of Fisher's linear discriminant,

which is a commonly used method for supervised classification learning. It replaces the supervisor with the simple heuristics that two temporally adjacent samples are likely to be from the same class. Furthermore, the capability of SFA as a possible readout mechanism is analyzed: computer simulations of a generic cortical microcircuit model demonstrate that readouts trained with SFA are able to learn to detect repeating firing patterns within a stream of spike trains with the same firing statistics, as well as to discriminate between the network responses to different spoken digits in an unsupervised manner.

Finally, in Chapter 4 novel experimental evidence for the liquid computing model is presented. We analyzed the activity of individual neurons in primary auditory cortex (A1) of awake ferrets to what extent their responses to the current sound are influenced by the immediate history of auditory stimulation. We directly estimated the mutual information between the responses and both current and preceding sounds and compared this value to the amount of information that could be extracted by linear classifiers. This revealed that many neurons conveyed a significant amount of information simultaneously about currently and previously played tones, and that most of this information could be extracted by linear classifiers. Moreover, the neural response provided a nonlinear combination of previously arrived stimuli and made this information available to a linear decoder. These observations, that sequentially arriving stimulus information is integrated over time and superimposed in a non-linear way into the neural responses at one point in time, are both predictions of the liquid computing model. For the first time such evidence has been found in awake animals.

Chapter 2

Spiking neurons can learn to solve information bottleneck problems and to extract independent components

Contents

In this Chapter it is shown how two unsupervised learning mechanisms that are based on information theoretic principles, information bottleneck optimization and independent component analysis, can be implemented using biologically realistic neuron models. This Chapter is based on the paper Spiking neurons can learn to solve information bottleneck problems and to extract independent components *by Stefan Klampfl, Robert Legenstein, and Wolfgang Maass (*Neural Computation*, 21(4):911-959, 2009). RL did the derivation of the simplified rate based rule and helped with its analysis.*

Independent component analysis (or blind source separation) is assumed to be an essential component of sensory processing in the brain and could provide a less redundant representation about the external world. Another powerful processing strategy is the optimization of internal representations according to the information bottleneck method. This method would allow to extract preferentially those components from high-dimensional sensory input streams that are related to other information sources, such as internal predictions or proprioceptive feedback. However there exists a lack of models that could explain how spiking neurons could

learn to execute either of these two processing strategies. We show in this Chapter how stochastically spiking neurons with refractoriness could in principle learn in an unsupervised manner to carry out both information bottleneck optimization and the extraction of independent components. We derive suitable learning rules, which extend the well known BCM-rule, from abstract information optimization principles. These rules will simultaneously keep the firing rate of the neuron within a biologically realistic range.

2.1 Introduction

The Information bottleneck (IB) approach and independent component analysis (ICA) have both attracted substantial interest as general principles for unsupervised learning (Tishby et al., 1999; Hyvärinen et al., 2001). A hope has been, that they might also help us to understand strategies for unsupervised learning in biological systems. However it has turned out to be quite difficult to establish links between known learning algorithms that have been derived from these general principles, and learning rules that could possibly be implemented by synaptic plasticity of a spiking neuron. Fortunately, in a simpler context a direct link between an abstract information-theoretic optimization goal and a rule for synaptic plasticity has recently been established (Toyoizumi et al., 2005). The resulting rule for the change of synaptic weights in (Toyoizumi et al., 2005) maximizes the mutual information between pre- and postsynaptic spike trains, under the constraint that the postsynaptic firing rate stays close to some target firing rate. We show in this thesis, that this approach can be extended to situations where simultaneously the mutual information between the postsynaptic spike train of the neuron and other signals (such as for example the spike trains of other neurons) has to be minimized (see Figure 2.1). This opens the door to the exploration of learning rules for information bottleneck analysis and independent component extraction with spiking neurons that would be optimal from a theoretical perspective.

The information bottleneck method (Tishby et al., 1999) is a recently developed information-theoretic approach that tries to compress information about a data variable X, while at the same time preserving as much information as possible about a *relevant* (target) variable Y, i.e., it aims at selecting a compact representation $\tilde{X}$ of the data X. That is, information that X provides about Y is squeezed through a "bottleneck" of the compressed variable $\tilde{X}$. There is a trade-off between compression (low mutual information between $\tilde{X}$ and X) and preserving relevant information (high mutual information between $\tilde{X}$ and Y). That is, one usually maximizes $-I(\tilde{\mathbf{X}};\mathbf{X}) + \beta I(\tilde{\mathbf{X}};\mathbf{Y})$ with some trade-off parameter β, where $I(\mathbf{U};\mathbf{V})$ denotes the mutual information between random variables U and V. In this approach, we interpret the input spike trains X^K to a neuron as the data X, the output spike train Y_1^K as the compact representation $\tilde{X}$ of X, and the relevant variable Y as a "target" spike train Y_2^K (or several target spike trains $Y_2^K, Y_3^K, ...$) (see Figure 2.1A).

Independent component analysis (ICA) (Hyvärinen et al., 2001) is another well-known statistical technique for decomposing complex data into statistically inde-

pendent parts, thereby providing a less redundant representation. In our approach, we minimize the mutual information between the output spike trains Y_1^K and Y_2^K of two neurons receiving the same input X^K. Simultaneously we want both neurons to extract meaningful information by maximizing the mutual information between the inputs X^K and the output spike train Y_i^K of both neurons $i = 1, 2$ (see Figure 2.1B).

We review in section 2.2 the neuron model and learning rule from (Toyoizumi et al., 2005). We show in section 2.3 how this learning rule can be extended so that it not only maximizes mutual information with some given spike trains and keeps the output firing rate within a desired range, but simultaneously minimizes mutual information with other spike trains, or other time-varying signals. In section 2.4 we analyze the learning strategies of the resulting learning rules, and relate them to the classical (Bienenstock et al., 1982) and generalized (Toyoizumi et al., 2005) Bienenstock-Cooper-Munro (BCM) rule. Applications to concrete information bottleneck tasks are discussed in section 2.5. Because of the many different types of target signals that might be relevant in a biological system, we do not model the way how such target signals might affect the synapse or neuron under consideration, but rather use it as an abstract signal in the learning rule. In section 2.6 we show that a modification of this learning rule allows a spiking neuron to extract information from its input spike trains that is independent from the information extracted by another neuron. Moreover, we present an approximation of the learning rule which indicates how the learning rule might possibly be implemented in a biologically realistic circuit.

2.2 Neuron model and a basic learning rule

We use the neuron model from (Toyoizumi et al., 2005), which is a stochastically spiking neuron model with refractoriness, where the probability of firing in each time step depends on the current membrane potential and the time since the last output spike. It is convenient to formulate the model in discrete time with step size Δt. The total membrane potential of a neuron i at time step $t^k = k\Delta t$ is given by

$$u_i(t^k) = u_r + \sum_{j=1}^{N} \sum_{n=1}^{k} w_{ij} \varepsilon(t^k - t^n) x_j^n, \tag{2.1}$$

where $u_r = -70\text{mV}$ is the resting potential and w_{ij} is the weight of the synapse from the presynaptic neuron j ($j = 1, \ldots, N$). An input spike train at synapse j is described up to the k-th time step by a sequence $X_j^k = (x_j^1, x_j^2, \ldots, x_j^k)$ of zeros (no spike) and ones (spike). Each presynaptic spike at time t^n ($x_j^n = 1$) evokes a postsynaptic potential (PSP) with exponential by decaying time course $\varepsilon(t - t^n) = U_{PSP} e^{-(t-t_n)/\tau_m}$ for $t \geq t_n$ with time constant $\tau_m = 10\text{ms}$ and PSP amplitude $U_{PSP} = 1\text{mV}$. The probability ρ_i^k of the firing of neuron i at time step t^k is then given by

$$\rho_i^k = 1 - \exp[-g(u_i(t^k)) R_i(t^k) \Delta t] \approx g(u_i(t^k)) R_i(t^k) \Delta t, \tag{2.2}$$

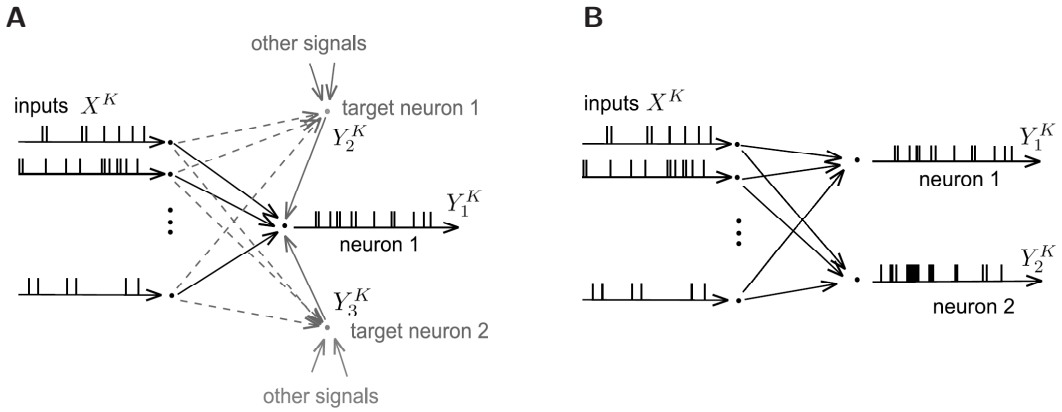

Figure 2.1: **Different learning situations analyzed in Chapter 2.** (**A**) In an information bottleneck task the learning neuron (neuron 1) wants to maximize the mutual information between its output Y_1^K and the activity of one or several target neurons $Y_2^K, Y_3^K, \ldots$ (which can be functions of the inputs X^K and/or other external signals), while at the same time keeping the mutual information between the inputs X^K and the output Y_1^K as low as possible (and its firing rate within a desired range). Thus the neuron should learn to extract from its high-dimensional input those aspects that are related to these target signals. This setup is discussed in sections 2.3-2.5. (**B**) Two neurons receiving the same inputs X^K from a common set of presynaptic neurons both learn to maximize information transmission, and simultaneously to keep their outputs Y_1^K and Y_2^K statistically independent. Such extraction of independent components from the input is described in section 2.6.

where the refractory variable

$$R_i(t) = \frac{(t - \hat{t}_i - \tau_{abs})^2}{\tau_{refr}^2 + (t - \hat{t}_i - \tau_{abs})^2} \Theta(t - \hat{t}_i - \tau_{abs}) \tag{2.3}$$

assumes values in $[0, 1]$ and depends on the last firing time $\hat{t}_i$ of neuron i (see Figure 2.2B). The absolute refractory period $\tau_{abs} = 3$ms is the time period after a firing during which no spike can occur; in the relative refractory time $\tau_{refr} = 10$ms it is hard, but not impossible, to emit an action potential. The Heaviside step function Θ takes a value of 1 for non-negative arguments and 0 otherwise. The gain function

$$g(u) = r_0 \log \left\{ 1 + \exp \left[\frac{u - u_0}{\Delta u} \right] \right\} \tag{2.4}$$

is a smooth increasing function of the membrane potential u (see Figure 2.2A; $u_0 = -65$mV, $\Delta u = 2$mV, $r_0 = 11$Hz). The approximation in (2.2) is valid for sufficiently small Δt ($\rho_i^k \ll 1$). The function $g(u)$ implements a stochastic threshold around u_0; below u_0 it goes to 0, above u_0 it increases linearly with the membrane potential (with slope $r_0/\Delta u$). Note that due to refractoriness the output firing rate of the neuron cannot be made arbitrarily high. For a neuron model without refractoriness (see section 2.3.2) one has to formalize an upper bound on the firing rate of the neuron in a different way. For that we choose as in (Toyoizumi et al.,

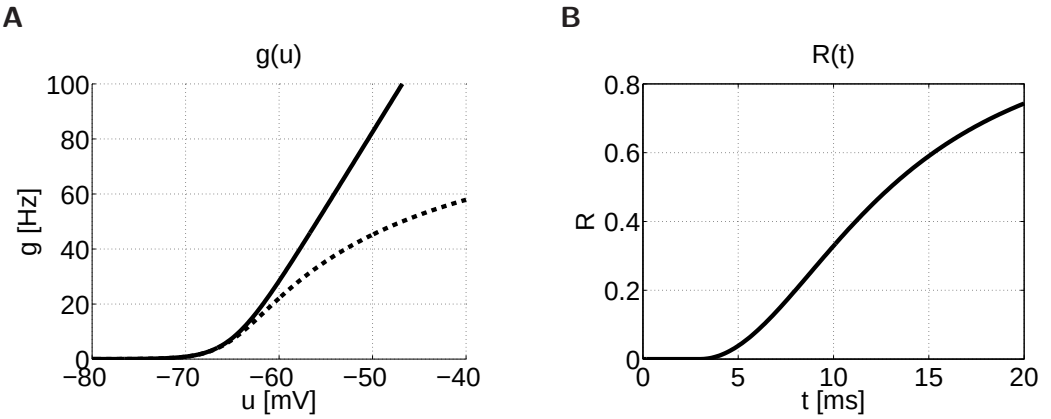

Figure 2.2: **Characteristic functions of the neuron model (see equation (2.2)).** (**A**) Gain function $g(u)$ (solid; see equ. (2.4)) and $g_{alt}(u)$ (dashed; see equ. (2.5)) as a function of the membrane potential u (plotted for $u_0 = -65$mV, $\Delta u = 2$mV, $r_0 = 11$Hz, $g_{max} = 100$Hz). (**B**) Refractory variable $R(t)$ as a function of the time $t - \hat{t}$ since the last postsynaptic spike (plotted for $\hat{t} = 0$, and for an absolute refractory period $\tau_{abs} = 3$ms, relative refractory time $\tau_{refr} = 10$ms).

2005) an alternative gain function

$$g_{alt}(u) = \left[\frac{1}{g_{max}} + \frac{1}{g(u)}\right]^{-1}, \tag{2.5}$$

with a maximum rate of $g_{max} = 100$Hz (see Figure 2.2A).

This model from (Toyoizumi et al., 2005) is a special case of the spike-response model, and with a refractory variable $R(t)$ that depends only on the time since the last postsynaptic event it has renewal properties (Gerstner and Kistler, 2002). The output of neuron i at the k-th time step is denoted by a variable y_i^k that assumes the value 1 if a postsynaptic spike occurs and 0 otherwise. A specific spike train up to the k-th time step is written as $Y_i^k = (y_i^1, y_i^2, \ldots, y_i^k)$.

The information transmission between an ensemble of input spike trains $\mathbf{X}^K$ and the output spike train $\mathbf{Y}_i^K$ of total duration $K\Delta t$ can be quantified by the mutual information[1] (Cover and Thomas, 1991)

$$I(\mathbf{X}^K; \mathbf{Y}_i^K) = \sum_{X^K, Y_i^K} P(X^K, Y_i^K) \log \frac{P(Y_i^K|X^K)}{P(Y_i^K)}. \tag{2.6}$$

The idea in (Toyoizumi et al., 2005) was to maximize the quantity $I(\mathbf{X}^K; \mathbf{Y}_i^K) - \gamma D_{KL}(P(Y_i^K)||\tilde{P}(Y_i^K))$, where

$$D_{KL}(P(Y_i^K)||\tilde{P}(Y_i^K)) = \sum_{Y_i^K} P(Y_i^K) \log \frac{P(Y_i^K)}{\tilde{P}(Y_i^K)} \tag{2.7}$$

[1] We use boldface letters ($\mathbf{X}^k$) to distinguish random variables from specific realizations (X^k).

denotes the Kullback-Leibler divergence (Cover and Thomas, 1991) between the actual distribution $P(Y_i^K)$ and a given target distribution $\tilde{P}(Y_i^K)$. The inclusion of this second term imposes the additional constraint that the firing statistics $P(Y_i)$ of the neuron i should stay as close as possible to a target distribution $\tilde{P}(Y_i)$. This distribution was chosen in (Toyoizumi et al., 2005) to yield a constant target firing rate $\tilde{g}$. An online learning-rule performing gradient ascent on this quantity was derived in (Toyoizumi et al., 2005) for the weight w_{ij} of neuron i:

$$\frac{dw_{ij}(t)}{dt} = \alpha C_{ij}(t) B_i^{post}(t,\gamma), \tag{2.8}$$

which consists of the "correlation term" C_{ij} and the "postsynaptic term" B_i^{post} (Toyoizumi et al., 2005). The term C_{ij} measures coincidences between postsynaptic spikes at neuron i and PSPs generated by presynaptic action potentials arriving at synapse j,

$$\frac{dC_{ij}(t)}{dt} = -\frac{C_{ij}(t)}{\tau_C} + \sum_l \varepsilon(t - t_j^{(l)}) \frac{g'(u_i(t))}{g(u_i(t))} [\delta(t - \hat{t}_i) - g(u_i(t)) R_i(t)], \tag{2.9}$$

with time constant $\tau_C = 1$s, $\delta(t)$ being the Dirac-δ function, and $g'(u_i(t))$ denoting the derivative of g with respect to u. The term

$$\begin{aligned} B_i^{post}(t,\gamma) = \delta(t - \hat{t}_i) \log \left[\frac{g(u_i(t))}{\bar{g}_i(t)} \left(\frac{\tilde{g}}{\bar{g}_i(t)} \right)^{\gamma} \right] \\ - R_i(t)[g(u_i(t)) - (1+\gamma)\bar{g}_i(t) + \gamma \tilde{g}] \end{aligned} \tag{2.10}$$

compares the current firing rate $g(u_i(t))$ with its average firing rate[2] $\bar{g}_i(t)$, and simultaneously the running average $\bar{g}_i(t)$ with the constant target rate $\tilde{g}$. The second argument indicates that this term also depends on the optimization parameter γ.

2.3 Information-theoretic principles provide learning rules for more complex learning goals

We extend the learning rule presented in the previous section to a more complex scenario, where the mutual information between the output spike train Y_1^K of the learning neuron (neuron 1) and some target spike trains Y_l^K $(l > 1)$ has to be maximized, while simultaneously minimizing the mutual information between the inputs X^K and Y_1^K. Obviously this is the generic IB scenario applied to spiking neurons (see Figure 2.1A). A learning rule for extracting independent components with spiking neurons (see section 2.6) can be derived in a similar manner, by just switching the signs of the first two terms in the objective function (2.11). In this section we derive two online learning rules, a spike-based and a simplified rate-based learning rule, for this information bottleneck task.

[2]The rate $\bar{g}_i(t) = \langle g(u_i(t)) \rangle_{\mathbf{X}|Y_i}$ denotes an expectation of the firing rate over the input distribution given the postsynaptic history and is implemented as a running average with an exponential time window (with a time constant of 10s).

2.3.1 Spike-based learning rule

For simplicity, we consider the case of an IB optimization for only one target spike train Y_2^K, and derive an update rule for the synaptic weights w_{1j} of neuron 1. The quantity to maximize is therefore

$$L = -I(\mathbf{X}^K; \mathbf{Y}_1^K) + \beta I(\mathbf{Y}_1^K; \mathbf{Y}_2^K) - \gamma D_{KL}(P(Y_1^K)||\tilde{P}(Y_1^K)), \tag{2.11}$$

where β and γ are optimization constants. To maximize this objective function, we derive the weight change Δw_{1j}^k during the k-th time step by gradient ascent on (2.11), assuming that the weights w_{1j} can change between some bounds $0 \leq w_{1j} \leq w_{max}$ (we assume $w_{max} = 1$ throughout this Chapter).

Now we have to calculate the gradient of L with respect to the weights of the learning neuron, w_{1j}. Note that all three terms of (2.11) implicitly depend on w_{1j} because the output distribution $P(Y_1^K)$ changes if we modify the weights w_{1j}. Since the first and the last term of (2.11) have already been considered (up to the sign) in (Toyoizumi et al., 2005), we will concentrate here on the middle term

$$L_{12} := \beta I(\mathbf{Y}_1^K; \mathbf{Y}_2^K) = \beta \sum_{Y_1^K, Y_2^K} P(Y_1^K, Y_2^K) \log \frac{P(Y_1^K, Y_2^K)}{P(Y_1^K)P(Y_2^K)} \tag{2.12}$$

and denote the contribution of the gradient of L_{12} to the total weight change Δw_{1j}^k in the k-th time step by $\Delta \tilde{w}_{1j}^k$. One can proceed here also similarly as in (Toyoizumi et al., 2005), but some additional aspects have to be taken into account.

Up to now we have considered only spike trains of length $K\Delta t$ in (2.11) and (2.12). In order to get an expression for the weight change in a specific time step k, $\Delta \tilde{w}_{1j}^k$, we have to calculate the contribution of this time bin to the objective function L_{12}. According to the chain rule of information theory (Cover and Thomas, 1991), we can write the probabilities $P(Y_i^K)$ and $P(Y_1^K, Y_2^K)$ occurring in (2.12) as products over the probability distributions of individual time bins given the corresponding postsynaptic histories, i.e., $P(Y_i^K) = \prod_{k=1}^K P(y_i^k | Y_i^{k-1})$ and $P(Y_1^K, Y_2^K) = \prod_{k=1}^K P(y_1^k, y_2^k | Y_1^{k-1}, Y_2^{k-1})$. As a consequence, we can express the middle term L_{12} in (2.11) as a sum over the contributions of individual time bins, $L_{12} = \sum_{k=1}^K \Delta L_{12}^k$, with

$$\Delta L_{12}^k = \left\langle \beta \log \frac{P(y_1^k, y_2^k | Y_1^{k-1}, Y_2^{k-1})}{P(y_1^k | Y_1^{k-1}) P(y_2^k | Y_2^{k-1})} \right\rangle_{\mathbf{X}^k, \mathbf{Y}_1^k, \mathbf{Y}_2^k}. \tag{2.13}$$

Hence, ΔL_{12}^k reflects the statistical dependence between the binary variables y_1^k and y_2^k, given the postsynaptic histories Y_1^{k-1} and Y_2^{k-1}. An evaluation of the probabilities used in (2.13) can be found in Appendix B.1.1.

The weight change $\Delta \tilde{w}_{1j}^k$ in each time step k is then proportional to the gradient of this expression ΔL_{12}^k with respect to the weights w_{1j},

$$\Delta \tilde{w}_{1j}^k = \alpha \frac{\partial \Delta L_{12}^k}{\partial w_{1j}}, \tag{2.14}$$

where $\alpha > 0$ denotes the learning rate. Under the assumption of small Δt (we choose $\Delta t = 1$ms throughout the simulations), evaluation of the gradient (2.14) yields (for a detailed derivation see Appendix B.1.2)

$$\Delta\tilde{w}_{1j}^k = \alpha \left\langle C_{1j}^k \beta F_{12}^k \right\rangle_{\mathbf{X}^k, \mathbf{Y}_1^k, \mathbf{Y}_2^k} \ . \tag{2.15}$$

The term in the parentheses of (2.15) consists of two factors. The first factor is a correlation term C_{1j}^k as in (Toyoizumi et al., 2005),

$$C_{1j}^k = C_{1j}^{k-1}\left(1 - \frac{\Delta t}{\tau_C}\right) + \sum_{n=1}^{k} \varepsilon(t^k - t^n) x_j^n \frac{g'(u_1(t^k))}{g(u_1(t^k))} \left[y_1^k - \rho_1^k\right]. \tag{2.16}$$

which counts the coincidences between postsynaptic spikes ($y_1^k = 1$) and the time course of PSPs generated by presynaptic spikes ($x_j^n = 1$) in an exponential time window with time constant $\tau_C = 1$s. The term $g'(u_i(t))$ denotes the derivative of $g(u)$ with respect to u and measures the sensitivity of the neuron for changes in the membrane potential.

The second factor measures the momentary statistical dependence between the outputs y_1^k and y_2^k,

$$\begin{aligned} F_{12}^k = {} & y_1^k y_2^k \log \frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)\bar{g}_2(t^k)} - y_1^k(1-y_2^k) R_2(t^k)\Delta t \left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)} - \bar{g}_2(t^k)\right] - \\ & - (1-y_1^k) y_2^k R_1(t^k)\Delta t \left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)} - \bar{g}_1(t^k)\right] + \\ & + (1-y_1^k)(1-y_2^k) R_1(t^k) R_2(t^k) (\Delta t)^2 \left[\bar{g}_{12}(t^k) - \bar{g}_1(t^k)\bar{g}_2(t^k)\right]. \end{aligned} \tag{2.17}$$

Here, $\bar{g}_i(t^k) = \langle g(u_i(t^k))\rangle_{\mathbf{X}^k|Y_i^{k-1}}$ denotes the average firing rate of neuron i and $\bar{g}_{12}(t^k) = \langle g(u_1(t^k)) g(u_2(t^k))\rangle_{\mathbf{X}^k|Y_1^{k-1}, Y_2^{k-1}}$ denotes the average product of firing rates of both neurons. Both quantities are implemented online as running exponential averages with a time constant of 10s. Note that F_{12}^k depends directly on the relationship between the joint probability of firing, which is represented by $\bar{g}_{12}(t^k)$, and the product of the individual firing probabilities given by $\bar{g}_1(t^k)\bar{g}_2(t^k)$.

Yet, the weight change (2.15) is still given by an average over the distributions of spike trains X^k, Y_1^k, Y_2^k up to time step k and cannot be implemented as an online rule in this way. However, under the assumption of a small learning rate α we can approximate the expectation $\langle\cdot\rangle_{\mathbf{X}^k, \mathbf{Y}_1^k, \mathbf{Y}_2^k}$ in (2.13) by averaging over a single long trial. Considering now all three terms in (2.11) we finally arrive at an online rule for maximizing L,

$$\frac{\Delta w_{1j}^k}{\Delta t} = -\alpha C_{1j}^k \left[B_1^k(-\gamma) - \beta \Delta t B_{12}^k\right]. \tag{2.18}$$

The term C_{1j}^k (2.16) is sensitive to correlations between the output of the neuron and its presynaptic input at synapse j ("correlation term") and the terms B_1^k and B_{12}^k characterize the postsynaptic state of the neuron ("postsynaptic terms"). Typical time courses of these terms are shown in Figure 2.3.

This learning rule is thus an extension to the generalized BCM rule for spiking neurons (Toyoizumi et al., 2005): The term $B_1^k(-\gamma)$ is given by

$$\begin{aligned} B_1^k(-\gamma) = \frac{y_1^k}{\Delta t} \log \left[\frac{g(u_1(t^k))}{\bar{g}_1(t^k)} \left(\frac{\bar{g}_1(t^k)}{\tilde{g}} \right)^{\gamma} \right] \\ - (1-y_1^k) R_1(t^k) \left[g(u_1(t^k)) - (1-\gamma)\bar{g}_1(t^k) - \gamma \tilde{g} \right], \end{aligned} \tag{2.19}$$

and has been described together with C_{1j}^k in the previous section (these terms are discrete-time versions of $C_{1j}(t)$ (2.9) and $B_1^{post}(t,\gamma)$ (2.10), respectively[3]). Our learning rule contains an extra term $B_{12}^k = F_{12}^k/(\Delta t)^2$ that is sensitive to the statistical dependence between the output spike train of the neuron and the target signal. It is given by

$$\begin{aligned} B_{12}^k = \frac{y_1^k y_2^k}{(\Delta t)^2} \log \frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)\bar{g}_2(t^k)} - \frac{y_1^k}{\Delta t}(1-y_2^k) R_2(t^k) \left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)} - \bar{g}_2(t^k) \right] \\ - \frac{y_2^k}{\Delta t}(1-y_1^k) R_1(t^k) \left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)} - \bar{g}_1(t^k) \right] \\ + (1-y_1^k)(1-y_2^k) R_1(t^k) R_2(t^k) \left[\bar{g}_{12}(t^k) - \bar{g}_1(t^k)\bar{g}_2(t^k) \right]. \end{aligned} \tag{2.20}$$

This term basically compares the average product of firing rates $\bar{g}_{12}$ (which corresponds to the joint probability of spiking) with the product of average firing rates $\bar{g}_1\bar{g}_2$ (representing the probability of independent spiking). In this way, it measures the momentary mutual information between the output of the neuron and the target spike train. B_{12}^k consists of four terms, one for each firing state of the two neurons. The first term produces a peak when both neurons fire in the same time step (cf. positive peak in bottom trace of Figure 2.3). The second and third term result in peaks when only one neuron is active (negative peaks in Figure 2.3). Note that these terms additionally depend on the refractory state of the other neuron: in case of almost coincident spikes the second event has no influence due to the refractoriness of the other neuron which has spiked just before. In other words, the learning rule distinguishes between the two cases whether a neuron does not spike because of refractoriness or because of a low firing rate; only in the latter case this has an influence. Finally, the fourth term of (2.20) results in small fluctuations of B_{12}^k in between firing events, and depends on the refractoriness of both neurons and the difference between $\bar{g}_{12}$ and $\bar{g}_1\bar{g}_2$. Note, however, that the actual sign of B_{12}^k depends on the recent firing histories of the two neurons, e.g., if the two spike trains have recently been correlated, $\bar{g}_{12}$ is larger than $\bar{g}_1\bar{g}_2$ (as is the case in Figure 2.3). Furthermore, the actual weight change depends according to (2.18) on an interplay between both the postsynaptic terms B_1^k and B_{12}^k and the correlation term C_{1j}^k.

Note that during the duration of an EPSP caused by an input spike there is an increased probability of generating an output spike (Kempter et al., 1999). If two neurons share the same input, they will then have a correlated spiking probability.

[3]The argument of B_1^k, $-\gamma$, is different from the second argument in (2.8), γ, because the the term $I(\mathbf{X}^K;\mathbf{Y}_1^K)$ enters the objective function (2.11) with a different sign, whereas the constraint with the KL-divergence enters with the same sign.

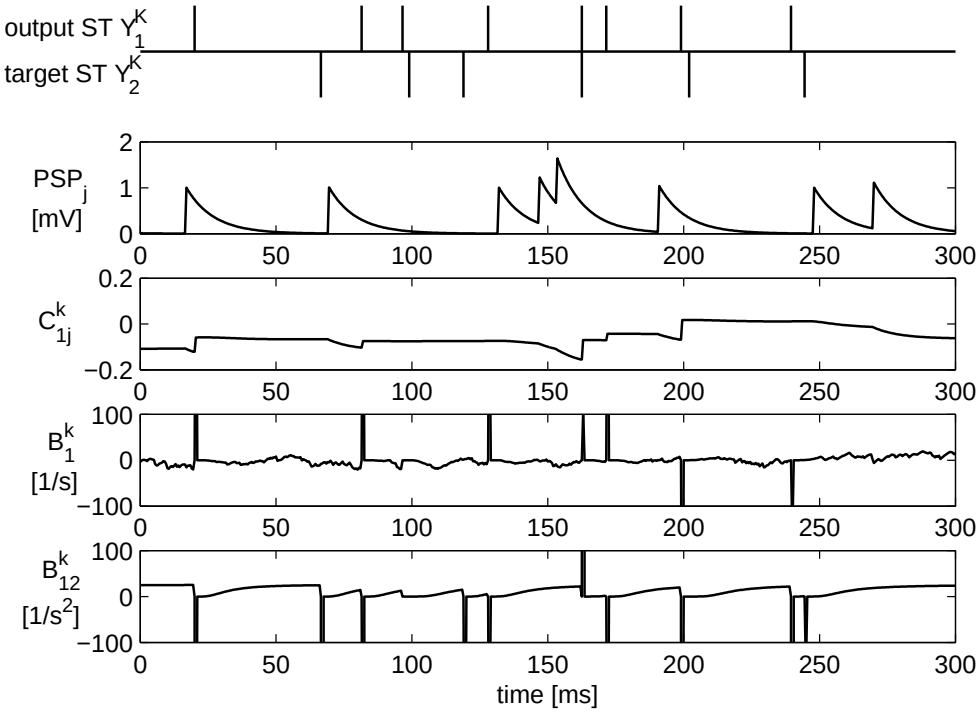

Figure 2.3: **Visualization of the impact of the three terms in learning rule** (2.18)**.** From top to bottom: instances of an output spike train Y_1^K and a target spike train Y_2^K of length 300ms, the time course of the PSP $\sum_n \varepsilon(t^k - t^n)x_j^n$ during that time at a single synapse j, of the correlation term C_{1j}^k (2.16) for the input at this synapse j, and of the postsynaptic terms B_1^k (2.19) and B_{12}^k (2.20). While the term B_1^k has peaks only for spikes in the output spike train Y_1^K, the term B_{12}^k has additional peaks at times of action potentials in the target spike train Y_2^K. Their amplitude and sign depend on the momentary statistical dependence of the recent histories of both spike trains.

This effect of the EPSP is captured by the correlation term C_{ij}^k (2.16), which is sensitive to correlations between input and output spikes. It is increased if an input spike is accompanied by an output spike during the duration of the EPSP caused by that input spike. Note that the term B_{12}^k in (B.39) is multiplied with the term C_{ij}^k in the actual learning rule (2.18). The term B_{12}^k on its own is only sensitive to the mutual information between the binary variables y_1^k and y_2^k given their histories (estimated by the running averages of firing rates), regardless of how they have been generated.

In (2.18), in order to compensate the effect of a small Δt, the constant β has to be large enough for the term B_{12}^k to have an influence on the weight change. In the limit $\Delta t \to 0$ the value of β approaches infinity. One can overcome this problem by using instead of (2.11) an alternative objective function which includes the information rate $I(\mathbf{Y}_1^K; \mathbf{Y}_2^K)/\Delta t$ instead of the mutual information $I(\mathbf{Y}_1^K; \mathbf{Y}_2^K)$.

In this case the Δt on the right-hand side of (2.11) would cancel out, and the trade-off parameter β would become a constant of dimension s (time). However, in the following we use our original objective function (2.11) and analyze weight changes in discrete time with a fixed Δt.

2.3.2 Simplified rate-based learning rule

To gain more insight in the learning rule (2.18), we consider a simplified neuron model without refractoriness. The dynamics of this model are governed by equations (2.1) and (2.2) with $R_i(t) = 1$ (i.e., $\tau_{abs} = \tau_{refr} = 0$ms). As in (Toyoizumi et al., 2005) we use $g_{alt}(u)$ (2.5) for the gain function in order to pose an upper limit on the postsynaptic firing rate in the absence of refractoriness. In this rate model, the probability of spiking is independent of the postsynaptic history. Since there is no refractoriness, the postsynaptic rate ν_1^k at time t^k is given directly by the current value of $g_{alt}(u_1(t^k))$. It was shown in (Toyoizumi et al., 2005) that the update rule (2.8) resembles the BCM-rule (Bienenstock et al., 1982). Since we want to maximize here a different objective function (2.11), we expect an "anti-Hebbian BCM" rule with an additional term accounting for statistical dependencies between Y_1^K and Y_2^K.

With these simplifying assumptions above the learning rule (2.18) reduces to the following learning rule for a rate model (see Appendix B.1.4 for a detailed derivation):

$$\frac{\Delta w_{1j}^k}{\Delta t} = -\alpha \nu_j^{pre,k} f(\nu_1^k) \left\{ \log \left[\frac{\nu_1^k}{\bar{\nu}_1^k} \left(\frac{\bar{\nu}_1^k}{\tilde{g}} \right)^\gamma \right] - \beta \Delta t \left(\nu_2^k \log \left[\frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k} \right] - \bar{\nu}_2^k \left[\frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k} - 1 \right] \right) \right\}, \quad (2.21)$$

where the presynaptic rate at synapse j at time t^k is denoted by $\nu_j^{pre,k} = a \sum_{n=1}^k \varepsilon(t^k - t^n) x_j^n$ with a in units $(\mathrm{Vs})^{-1}$. The values $\bar{\nu}_1^k$, $\bar{\nu}_2^k$, and $\bar{\nu}_{12}^k$ are running averages of the output rate ν_1^k, the rate of the target signal ν_2^k and of the product of these values, $\nu_1^k \nu_2^k$, respectively. The function $f(\nu_1^k) = g'_{alt}(g_{alt}^{-1}(\nu_1^k))/a$ is proportional to the derivative of g_{alt} with respect to u, evaluated at the current membrane potential. It measures the momentary sensitivity of the output rate for changes of the membrane potential (see Figure 2.4A). This weight change approximates a gradient ascent for the objective function (2.11). The approximation is valid for small Δt (we choose $\Delta t = 1$ms in the simulations). Note that the factor β has to compensate a small Δt so that the second term has influence on the weight change. A detailed discussion of this rule is given in section 2.4.

2.4 Analysis of the resulting learning rules

In the previous section we have derived learning rules that minimize the information transmission of a neuron while simultaneously keeping the mutual information between the output and target spike trains as high as possible. Additionally we

Spike-based rule for the IB task:

Performing gradient ascent on L (2.11) yields an online learning rule for the weights of neuron 1, w_{1j}. The weight change Δw_{1j}^k at time $t^k = k\Delta t$ is given by

$$\frac{\Delta w_{1j}^k}{\Delta t} = -\alpha C_{1j}^k \left[B_1^k(-\gamma) - \beta \Delta t B_{12}^k \right] \tag{2.18}$$

with a learning rate $\alpha > 0$ and optimization parameters β and γ with values > 0.

The correlation term C_{1j}^k measures coincidences between postsynaptic spikes at neuron 1 and PSPs generated by presynaptic action potentials arriving at synapse j:

$$C_{1j}^k = C_{1j}^{k-1} \left(1 - \frac{\Delta t}{\tau_C}\right) + \sum_{n=1}^{k} \varepsilon(t^k - t^n) x_j^n \frac{g'(u_1(t^k))}{g(u_1(t^k))} \left[y_1^k - \rho_1^k \right] \tag{2.16}$$

τ_C	time constant of exponential correlation window
x_j^n	binary variable indicating a presynaptic spike at synapse j in the n-th time step
y_1^k	binary variable indicating an output spike of neuron 1 in the k-th time step
ρ_1^k	firing probability of neuron 1 in the k-th time step (2.2)
$\varepsilon(s)$	time course of PSP in response to a presynaptic spike at time $s = 0$
$g(u_1(t))$	gain function (2.4) evaluated at the value of the membrane potential $u_1(t)$ of neuron 1
$g'(u)$	derivative of $g(u)$ with respect to u

The term B_1^k is responsible for regulating the mutual information between input and output and maintaining the constant target firing rate for neuron 1:

$$\begin{aligned} B_1^k(\gamma) = \frac{y_1^k}{\Delta t} \log \left[\frac{g(u_1(t^k))}{\bar{g}_1(t^k)} \left(\frac{\tilde{g}}{\bar{g}_1(t^k)} \right)^{\gamma} \right] \\ - (1 - y_1^k) R_1(t^k) \left[g(u_1(t^k)) - (1+\gamma)\bar{g}_1(t^k) + \gamma \tilde{g} \right] \end{aligned} \tag{2.19}$$

$R_1(t^k)$	refractory variable (2.3) of neuron 1 at time t^k
$\bar{g}_1(t^k)$	running average of the postsynaptic firing rate $g(u_1(t^k))$ of neuron 1
$\tilde{g}$	constant target firing rate

Table 2.1: (continues on next page)

have imposed the constraint that the firing rate of the learning neuron should stay close to a constant target firing rate. These rules are summarized in Tables 2.1 and 2.2. The spike-based rule has been derived for a stochastically spiking neuron model with refractoriness; for the rate-based rule we considered a simplified neuron model without refractoriness, as in (Toyoizumi et al., 2005). In this section we interpret these rules and show how they relate to the classical BCM rule and to the generalized rule presented in (Toyoizumi et al., 2005).

2.4.1 Comparison of the simplified rule with the spike-based rule

Comparing the spike-based (2.18) and rate-based learning rule (2.21), we find that for both rules the weight change depends on the correlation of pre- and postsynaptic

The term B_{12}^k measures the mutual information between the output spike train Y_1^k of neuron 1 and and the target spike train Y_2^k:

$$\begin{aligned} B_{12}^k = &\frac{y_1^k y_2^k}{(\Delta t)^2} \log \frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)\bar{g}_2(t^k)} - \frac{y_1^k}{\Delta t}(1-y_2^k)R_2(t^k)\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)} - \bar{g}_2(t^k)\right] \\ &- \frac{y_2^k}{\Delta t}(1-y_1^k)R_1(t^k)\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)} - \bar{g}_1(t^k)\right] \\ &+ (1-y_1^k)(1-y_2^k)R_1(t^k)R_2(t^k)\left[\bar{g}_{12}(t^k) - \bar{g}_1(t^k)\bar{g}_2(t^k)\right] \end{aligned} \tag{2.20}$$

y_2^k	binary variable indicating a spike in the target spike train in the k-th time step
$R_2(t^k)$	refractory state of target spike train
$\bar{g}_2(t^k)$	running average of firing rate of target spike train
$\bar{g}_{12}(t^k)$	running average of the product between firing rates of the output and target spike train

Table 2.1: (continued) **Summary of the spike-based learning rule for the information bottleneck task derived in section 2.3.1.**

Simplified (rate-based) rule for the IB task:

For a simplified neuron model without refractoriness the spike-based rule (2.18) reduces to the following rate-based rule:

$$\begin{aligned} \frac{\Delta w_{1j}^k}{\Delta t} = -\alpha \nu_j^{pre,k} f(\nu_1^k) \Bigg\{ &\log\left[\frac{\nu_1^k}{\bar{\nu}_1^k}\left(\frac{\bar{\nu}_1^k}{\tilde{g}}\right)^\gamma\right] \\ &-\beta \Delta t \left(\nu_2^k \log\left[\frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k}\right] - \bar{\nu}_2^k \left[\frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k} - 1\right]\right)\Bigg\} \end{aligned} \tag{2.21}$$

α	learning rate
β, γ	optimization parameters
$\nu_j^{pre,k}$	presynaptic firing rate at synapse j at time t^k
$f(\nu_1^k)$	sensitivity of neuron 1 at its current firing state ν_1^k
ν_1^k	output firing rate of neuron 1 at time t^k
ν_2^k	firing rate of the target signal at time t^k
$\bar{\nu}_1^k$, $\bar{\nu}_2^k$	running averages of ν_1^k and ν_2^k
$\bar{\nu}_{12}^k$	running average of the product $\nu_1^k \nu_2^k$

Table 2.2: **Summary of the simplified (rate-based) learning rule for the information bottleneck task derived in section 2.3.2.**

activity, either via the correlation term C_{ij}^k or via the Hebbian term $\nu_j^{pre,k} f(\nu_1^k)$. In both cases, the influence of the postsynaptic activity on the weight change depends also on the current sensitivity of the neuron, which is expressed through the derivative of g with respect to u (see plot of $f(\nu_1^k)$ in Figure 2.4A). Furthermore, the first term in the curly brackets of (2.21) corresponds to the first term of $B_1^k(-\gamma)$ (2.10). This classical BCM-term is responsible for regulating the information transmission

of the neuron and for the homeostatic process that tries to maintain a constant target firing rate, via a sliding threshold of the postsynaptic activity, $\bar{\nu}_1^k$ (Toyoizumi et al., 2005). However, this term is augmented by an expression sensitive to the statistical dependence between the output of the neuron and the target signal (second line in (2.21) and B_{12}^k (2.20)). Here, the second line in (2.21) corresponds to the first two terms in (2.20). All the other terms in B_1^k and B_{12}^k can be neglected in the rate-based rule for small Δt (see derivation in Appendix B.1.4 and analogous derivation in (Toyoizumi et al., 2005)).

2.4.2 Interpretation of the simplified rule

To gain a better understanding of the derived learning rule, we analyze the rate-based rule (2.21) in more detail. The prefactor of (2.21), $\nu_j^{pre,k} f(\nu_1^k)$, is a nonlinear Hebbian term because the weight change does not depend on the postsynaptic activity ν_1^k directly, but only via the nonlinear function f. It is proportional to the impact of synapse j onto the membrane potential at time t^k times the sensitivity of the output rate on changes of the membrane potential at time t^k. This prefactor distributes the weight changes given by the terms in the curly brackets to the individual synapse j. Changes of strongly active synapses are larger than those of relatively silent ones. We can divide the term in the curly brackets into three functionally different parts. Each of these parts corresponds to the optimization of one of the terms in (2.11). The first part, $\log(\nu_1^k/\bar{\nu}_1^k)$, together with the prefactor $\nu_j^{pre,k} f(\nu_1^k)$, drives the optimization of mutual information between inputs and outputs (note that this part is combined with the second part in (2.21), which is discussed below). The second part, $\log(\bar{\nu}_1^k/\tilde{g})^\gamma$, accounts for homeostatic processes to stabilize the output rate. These two parts together with the prefactor introduce competition between the synapses and, as already noted in (Toyoizumi et al., 2005), they implement a BCM-like learning rule. The third part is given by the two terms of the second line of (2.21). These terms drive the maximization of mutual information between the output of the neuron Y_1^K and the target signal Y_2^K. We investigate this part of the update rule in more detail. The correlation between ν_1^k and ν_2^k is measured by

$$\phi := \frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k}, \tag{2.22}$$

which appears in both terms of the second line of (2.21). It has value 1 for uncorrelated firing rates, values > 1 for positive correlations and values < 1 for negative correlations (anti-correlations). To see how the second line of (2.21) depends on the ratio between ν_2^k and $\bar{\nu}_2^k$, we assume that $\bar{\nu}_2^k$ is constant and introduce $\zeta := \nu_2^k/\bar{\nu}_2^k$. Then, the second line of (2.21) is proportional to

$$\zeta \log(\phi) - (\phi - 1). \tag{2.23}$$

For $\nu_2^k = \bar{\nu}_2^k$, this function is negative if $\phi \neq 1$ and zero if $\phi = 1$ (dashed line in Figure 2.4B).

Suppose that the output of the neuron is positively correlated with the target signal ($\phi > 1$; see Figure 2.4B). Then, a firing rate ν_2^k of this target signal sufficiently

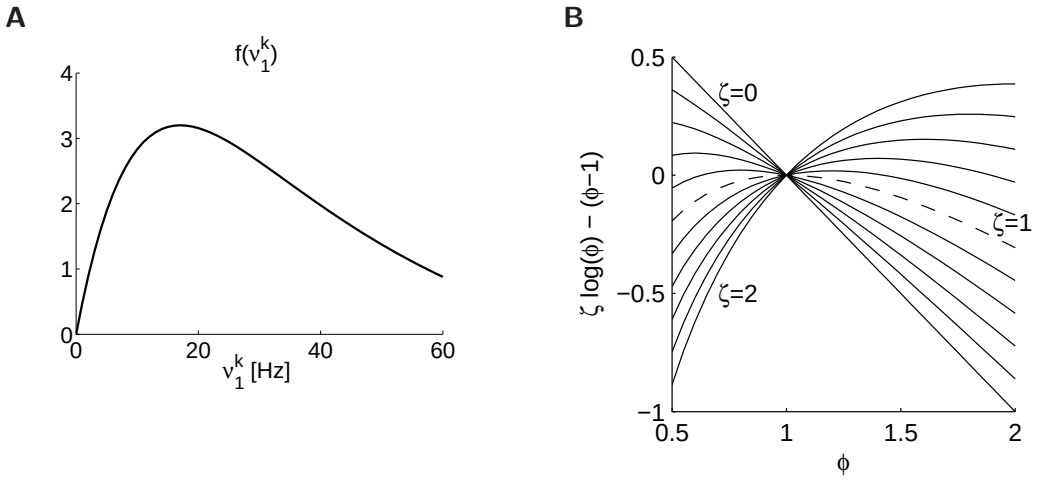

Figure 2.4: **Influence of specific terms of the rate-based rule** (2.21)**.** (**A**) Sensitivity function $f(\nu_1^k) = g'_{alt}(g_{alt}^{-1}(\nu_1^k))/a$ as a function of the postsynaptic firing rate ν_1^k with $a = 10^3(\mathrm{Vs})^{-1}$. (**B**) The influence of correlations between ν_1^k and ν_2^k (measured by $\phi = \bar{\nu}_{12}^k/(\bar{\nu}_1^k\bar{\nu}_2^k)$, see (2.22)) on the simplified rule for different ratios $\zeta = \nu_2^k/\bar{\nu}_2^k$. The plotted function captures the weight changes induced by the second line of (2.21). This function is zero for uncorrelated signals ($\phi = 1$). For correlated signals ($\phi > 1$), firing rates ν_2^k sufficiently above mean induce LTP. For anti-correlated signals ($\phi < 1$), firing rates ν_2^k sufficiently below mean induce LTP.

above mean (e.g., $\zeta = 2$) induces long term potentiation (LTP) in active synapses (i.e., synapses j with large $\nu_j^{pre,k}$). This will further increase the correlation between ν_1^k and ν_2^k for the encountered input. A firing rate ν_2^k of the target signal below mean ($\zeta < 1$) will induce long term depression (LTD) in active synapses. Again, this increases the correlation between ν_1^k and ν_2^k.

For anti-correlated signals ($\phi < 1$; see Figure 2.4B), firing rates ν_2^k sufficiently below mean (e.g., $\zeta = 0$) induce LTP in active synapses. This will increase the anti-correlation between ν_1^k and ν_2^k for the encountered input. Similarly, anti-correlation is increased for ν_2^k above mean, when LTD is induced in active synapses. Note that correlation and anti-correlation both contribute to the increase of mutual information.

2.4.3 Comparison with the BCM learning rue

To elucidate the relation to the classical Bienenstock-Cooper-Munro (BCM) learning rule (Bienenstock et al., 1982) we rewrite the simplified rule (2.21) as

$$\frac{\Delta w_{1j}^k}{\Delta t} = -\alpha\nu_j^{pre,k}\Phi(\nu_1^k, \nu_2^k), \tag{2.24}$$

where Φ is a two-dimensional function of the firing rates ν_1^k and ν_2^k,

$$\Phi(\nu_1^k, \nu_2^k) = f(\nu_1^k)\left\{\log\left[\frac{\nu_1^k}{\bar{\nu}_1^k}\left(\frac{\bar{\nu}_1^k}{\tilde{g}}\right)^\gamma\right] - \beta\Delta t\left[\nu_2^k\log\phi - \bar{\nu}_2^k(\phi-1)\right]\right\}, \tag{2.25}$$

with $\phi = \bar{\nu}_{12}^k/(\bar{\nu}_1^k\bar{\nu}_2^k)$. This function $\Phi(\nu_1^k, \nu_2^k)$ can be seen as an extension of the classical BCM synaptic modification function (Bienenstock et al., 1982; Toyoizumi et al., 2005) and is plotted in Figures 2.5A-2.5D for the special case that both average firing rates are equal to the constant target firing rate (i.e., $\bar{\nu}_1^k = \bar{\nu}_2^k = \tilde{g} =$ 20Hz), for four different values of the quotient ϕ.

Because of the anti-Hebbian nature of (2.24), values of Φ above 0 produce LTD. An analogous Hebbian learning rule for the extraction of independent components is derived in section 2.6. For such Hebbian learning rules, values of Φ above 0 produce LTP. One sees that for $\phi = 1$ (see Figure 2.5B) the second term in (2.25) vanishes, in which case Φ does not depend on ν_2^k and reduces to the classical BCM function in (Toyoizumi et al., 2005), where regimes of LTP and LTD are separated by a sliding threshold that depends in a nonlinear way on the running average of the postsynaptic rate $\bar{\nu}_1^k$. On the other hand, if $\phi \neq 1$ the value of Φ additionally depends on the current firing rate ν_2^k, which results in shifted versions of the BCM function where the balance between positive and negative domains varies as ν_2^k is changed from small to large values.

If $\phi < 1$, the signals are anti-correlated (see Figure 2.5A). In this case Φ is more negative for small values of ν_2^k and more positive for large values of ν_2^k. This means that for the anti-Hebbian learning rule (2.24), weights (and therefore also the firing rate ν_1^k) tend to increase for small ν_2^k, and decrease for large ν_2^k. Therefore the output of the neuron and the target signal get even more anti-correlated. Similarly, for correlated signals ($\phi > 1$, see Figures 2.5C and 2.5D) their correlation increases even further, since for small values of ν_2^k the output firing rate ν_1^k tends to decrease as well (due to positive values of Φ), whereas it grows for large ν_2^k (because of negative values of Φ). In both cases (correlated or anti-correlated signals) the statistical dependence between the output and the target signal increases, as should be the case for an IB-task.

2.4.4 Comparison with a previously proposed method for information bottleneck optimization

In the original formulation of the Information bottleneck method (Tishby et al., 1999) the data variable X should be compressed as much as possible by a quantization or representation $\tilde{X}$. At the same time, however, this compressed variable should capture as much information as possible about a relevance variable Y. There is a trade-off between compression and preserving meaningful information, leading to the following objective function to minimize:

$$L = I(\tilde{\mathbf{X}}; \mathbf{X}) - \beta I(\tilde{\mathbf{X}}; \mathbf{Y}), \tag{2.26}$$

where $\beta > 0$ is a trade-off parameter. If the joint distribution $P(X, Y)$ is given, the value of L depends only on the stochastic mapping[4] $P(\tilde{X}|X)$, because $\tilde{X}$ is independent of Y given X. For a given β the optimal solution which minimizes

[4]This means that the objective function L (2.26) can be written as a *functional* $\mathcal{L}[P(\tilde{X}|X)] = I(\tilde{\mathbf{X}}; \mathbf{X}) - \beta I(\tilde{\mathbf{X}}; \mathbf{Y})$ and minimizing (2.26) is equivalent to minimizing the functional $\mathcal{L}[P(\tilde{X}|X)]$ with respect to the conditional distribution $P(\tilde{X}|X)$ (Tishby et al., 1999).

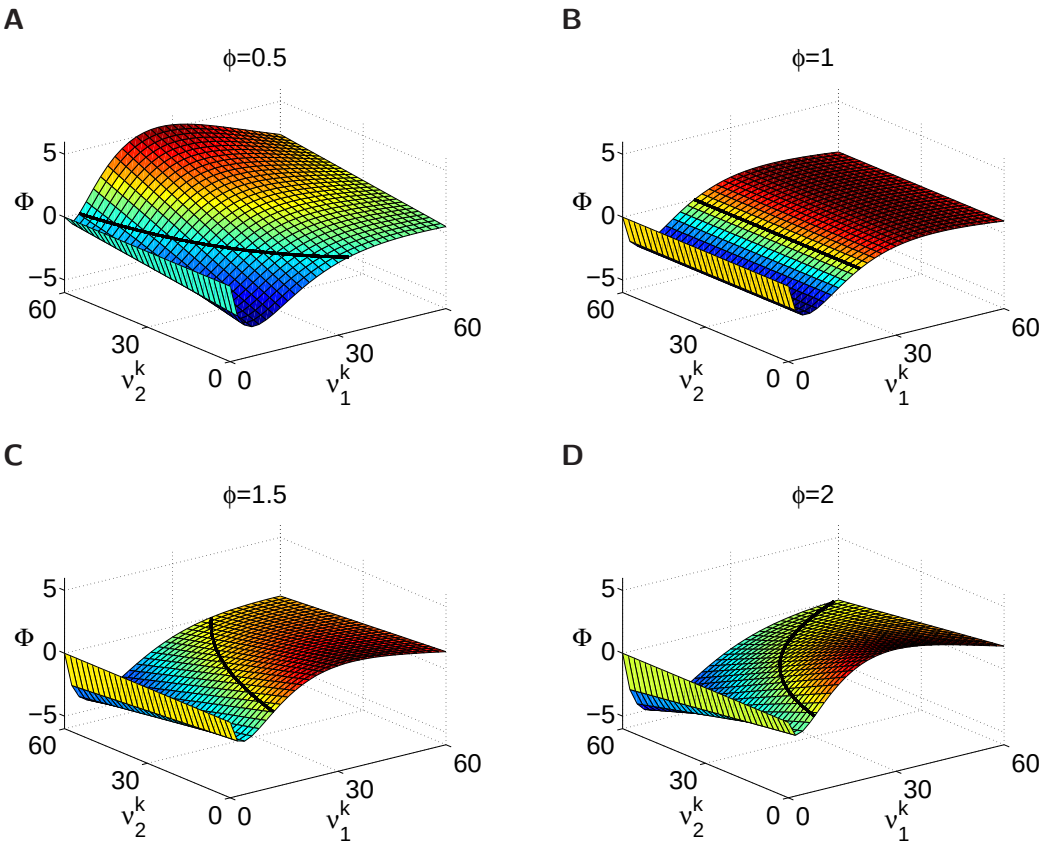

Figure 2.5: **Extension of the classical BCM rule into two dimensions.** Two-dimensional synaptic modification function $\Phi(\nu_1^k, \nu_2^k)$ (2.25) of the rate-based learning rule (2.21) as an extension of the classical BCM rule for $\bar{\nu}_1^k = \bar{\nu}_2^k = \tilde{g} = 20$Hz, $\beta = 50$, $\gamma = 1$, and different values of the quotient $\phi = \bar{\nu}_{12}^k/(\bar{\nu}_1^k\bar{\nu}_2^k)$, which measures the correlation between the output of the neuron and the target signal. The sliding threshold between LTP and LTD depends not only on the postsynaptic firing rate r_1^k, but also on the target signal r_2^k if both signals are correlated ($\phi > 1$) or anti-correlated ($\phi < 1$). ((**A**) $\phi = 0.5$, (**B**) $\phi = 1$, (**C**) $\phi = 1.5$, (**D**) $\phi = 2$). Note that Φ is reduced to a one-dimensional function (like in the classical BCM-rule) for $\phi = 1$ (see panel **B**). In each plot the solid black line indicates the transition from depression to potentiation ($\Phi = 0$).

(2.26) is given by (Tishby et al., 1999)

$$P(\tilde{X}|X) = \frac{P(\tilde{X})}{Z(X,\beta)} \exp\left[-\beta D_{KL}(P(Y|X)||P(Y|\tilde{X}))\right], \tag{2.27}$$

where $Z(X,\beta)$ is a normalization function. Note that equation (2.27) is implicit because both $P(\tilde{X})$ and $P(Y|\tilde{X})$ depend on $P(\tilde{X}|X)$, through

$$P(\tilde{X}) = \sum_X P(X)P(\tilde{X}|X) \tag{2.28}$$

and

$$P(Y|\tilde{X}) = \frac{1}{P(\tilde{X})} \sum_X P(X,Y)P(\tilde{X}|X). \tag{2.29}$$

The equations (2.27) to (2.29) can be solved iteratively with an extension of the Blahut-Arimoto (BA) algorithm, which is well-known from applications to problems from rate distortion theory and channel capacity calculations (Tishby et al., 1999; Cover and Thomas, 1991). This generalized Blahut-Arimoto algorithm performs alternating iterations over the distributions $P(\tilde{X}|X)$, $P(\tilde{X})$, and $P(Y|\tilde{X})$ and can be shown to converge to the optimal solution of (2.27) to (2.29) (Tishby et al., 1999). In the following we briefly discuss the relationship between this traditional IB algorithm and our learning rule for spiking neurons.

The traditional Information bottleneck approach has so far mainly been applied to discrete variables X, $\tilde{X}$ and Y, in a wide range of applications, see (Slonim, 2002) for a review and references. However, in the general theory there is no restriction on the type of these variables. In this thesis we apply the Information bottleneck principle to spike trains (see Figure 2.1A): The input spike trains X^K to the learning neuron correspond to the data variable X, the output spike train Y_1^K of this neuron represents the compressed variable $\tilde{X}$, and the target spike train Y_2^K specifies the relevant variable Y. This yields the following correspondence to the notation of (Tishby et al., 1999):

$$\begin{aligned} P(\tilde{X}|X) &\triangleq P(Y_1^K|X^K), \\ P(\tilde{X}) &\triangleq P(Y_1^K), \\ P(Y|\tilde{X}) &\triangleq P(Y_2^K|Y_1^K). \end{aligned}$$

In the traditional IB algorithms one usually specifies the trade-off parameter β and the joint distribution $P(X,Y)$ in advance. This is also the case for our experiments (see section 2.5) where we choose a particular statistics for the input and target spike trains, X^K and Y_2^K. Furthermore, both the BA algorithm and our learning rule search for the optimal distributions $P(\tilde{X}|X)$ and $P(Y_1^K|X^K)$, respectively. However, the compression achieved by the mapping $P(Y_1^K|X^K)$ is not modeled explicitly, but implicitly through the weights w_{1j} of the learning neuron. By updating these weights we successively adapt the stochastic input-output relationship given by $P(Y_1^K|X^K)$. Due to this modification of $P(Y_1^K|X^K)$ the distributions $P(Y_1^K) = \left\langle P(Y_1^K|X^K)\right\rangle_{\mathbf{X}^K}$ and $P(Y_2^K|Y_1^K) = P(Y_1^K, Y_2^K)/P(Y_1^K)$ change implicitly, whereas in the traditional IB algorithm the corresponding distributions $P(\tilde{X})$ and $P(Y|\tilde{X})$ are updated in a separate step. However, as in the Blahut-Arimoto algorithm, where the new value of $P(\tilde{X}|X)$ depends on the values of $P(\tilde{X})$ and $P(Y|\tilde{X})$, in our learning rule the adaptation of $P(Y_1^K|X^K)$, i.e., the change of the weights w_{1j}, depends on $P(Y_1^K)$ and $P(Y_2^K|Y_1^K)$ through the terms C_{1j}^k, B_1^k, and B_{12}^k of the learning rule (2.18).

More precisely, by comparing the current firing rate with its running average, the term B_1^k (2.19) depends on both the output distribution $P(Y_1^k)$ and the probability of the output given the input spike trains, $P(Y_1^k|X^k)$ (see also (Toyoizumi et al., 2005)). The distribution $P(Y_2^k|Y_1^k)$ influences the term B_{12}^k (2.20) since this

term compares the joint probability $P(Y_1^k, Y_2^k)$ with the independent distribution $P(Y_1^k)P(Y_2^k)$, or equivalently, $P(Y_2^k|Y_1^k)$ with $P(Y_2^k)$. Finally, both terms B_1^k and B_{12}^k are multiplied in the learning rule (2.18) with the correlation term C_{1j}^k, which can be written as the derivative of the logarithm of $P(Y_1^K|X^K)$ with respect to the weights w_{1j}, $C_{1j}^k = \frac{\partial}{\partial w_{1j}} \log P(Y_1^k|X^k)$ (see (2.16) and Appendix B.1.2).

Summarizing, the main difference between the previous Information bottleneck approach from (Tishby et al., 1999) and our special application to spiking neurons is that in our case the distribution under consideration, $P(Y_1^K|X^K)$, is parametrized by the weights w_{1j} of a spiking neuron (whereas in most IB algorithms, no special assumptions are made about the probability distributions to be optimized), and our learning rule is an online learning rule performing gradient ascent on the objective function. In the generalized Blahut-Arimoto algorithm the probability distributions are changed directly (e.g., by maintaining probability tables) and always converge to the optimal solution, whereas our learning rule can change them only implicitly by adapting the weights w_{1j}, and there is no guarantee that the global optimum is found. Another difference is that the Information bottleneck algorithm from (Tishby et al., 1999) is an offline algorithm that performs the optimization over the whole range of the random variables, whereas our algorithm is an online algorithm where the weights are adapted sequentially as the input and the target spike trains are presented to the neuron. In this sense our update rule can be viewed as a novel online learning approach to Information bottleneck optimization for a concrete parametrized instance of the problem.

2.5 Application to information bottleneck optimization

We use a setup as in Figure 2.1A where we want to maximize the information which the output Y_1^K of a learning neuron conveys about one or more target signals Y_2^K, Y_3^K, ... In the following simulations we let the neuron receive inputs X^K at $N = 100$ synapses, with weights randomly initialized at small values (from 0.10 to 0.12). Unless stated otherwise, we choose $\tilde{g} = 30$Hz for the target firing rate, and we use discrete time with $\Delta t = 1$ms.

2.5.1 Extracting a single rate modulation

In a first experiment we investigate how the spike-based learning rule (2.18) performs in a simple rate coding paradigm, i.e., the information is encoded in the firing rates of the spike trains. We divide the inputs into 4 groups of 25 synapses each. In the following, let $r_i(t)$ and $r_T(t)$ denote the firing rate of group i ($i = 1, \ldots, 4$) and of the target signal, respectively, at time t. Each input spike train is generated by an inhomogeneous Poisson process with common rate modulation within each group, however, the rate modulations for different groups are statistically independent (see Figure 2.6A). More precisely, for input group 1 (synapses 1 to 25) we choose a periodic rate modulation $r_1(t) = r_0 + A\sin(2\pi t/T)$ with $r_0 = 20$Hz, $A = 10$Hz, and $T = 500$ms. The rate of group 2 (synapses 26 to 50) is constant during intervals of 1s, each second a firing rate is chosen randomly out of the values 2Hz, 13Hz, 25Hz, 40Hz, and 50Hz. Synapses 51 to 75 (input group 3) receive a

rate that has a constant value of 2Hz, except that a burst is initiated at each time step with a probability of 0.0005. Thus there is a burst on average every 2s. The duration of a burst is chosen from a Gaussian distribution with mean 0.5s and SD 0.2s, the minimum duration is chosen to be 0.1s. During a burst the rate is set to 50Hz. Finally the remaining synapses (76 to 100; group 4) receive constant rate Poisson spike trains at 20Hz.

We generate the target spike train by an inhomogeneous Poisson process with the same rate modulation as the inputs of group 1, $r_1(t)$. In this case we expect that weights will grow only for the first group and remain depressed for the other inputs, since these are the only inputs that are not statistically independent from the target signal. However, Figure 2.6 shows that besides for group 1, strong weights are also developed for group 4, which is the uncorrelated constant rate Poisson input. This is because the neuron has to achieve a mean postsynaptic firing rate close to the constant target firing rate of 30Hz and uncorrelated Poisson spike trains with a constant rate are always statistically independent from any other spike train. Therefore, developing strong weights for this group of inputs does not increase the mutual information between input and output, which should be kept as low as possible. All other synapses are depressed because their inputs are statistically independent from the target signal. Moreover, Figure 2.6 shows that after learning the time course of the output rate modulation is similar to that of the target signal, therefore the neuron has learned to "represent" the target signal. Furthermore, the mutual information between input and output decreases, whereas the information as well as the correlation between the output and the target signal increases.

Further experiments show that one can also extract the rates of input groups 2 and 3, $r_2(t)$ and $r_3(t)$, if a correlated spike train is chosen as the target signal. However, it is not reasonable to take an uncorrelated fixed-rate Poisson spike train as the target spike train, since it does not contain mutual information with any of the inputs. Using such target a has the same effect as removing the target signal (see next experiment).

2.5.2 Extracting a time-varying combination of rate modulations

In the second experiment we consider a target signal that is only indirectly related to some of the inputs, and in addition this relationship varies over time. Again, the input is divided into 4 groups of 25 synapses each with different rate modulations. This time we use rates that are constant during random intervals and can take 5 different values, 2Hz, 13Hz, 25Hz, 40Hz, and 50Hz. The time during which the rate remains constant is drawn uniformly from the interval $[0\text{s}, 1\text{s}]$, and the value of the rate is also chosen uniformly among the 5 available values. The spike trains are generated from an inhomogeneous Poisson process with a rate modulation created with this method for each of the 4 input groups, independently from each other (see Figure 2.7A).

The rate of the target signal $r_T(t)$ is chosen to be a linear combination of the input rates. At the beginning, we set it to the mean between the rates of group 1 and 2, i.e., $r_T(t) = (r_1(t) + r_2(t))/2$, in order to test whether this suffices for triggering the increase of weights from input groups 1 and 2. To make the experiment more

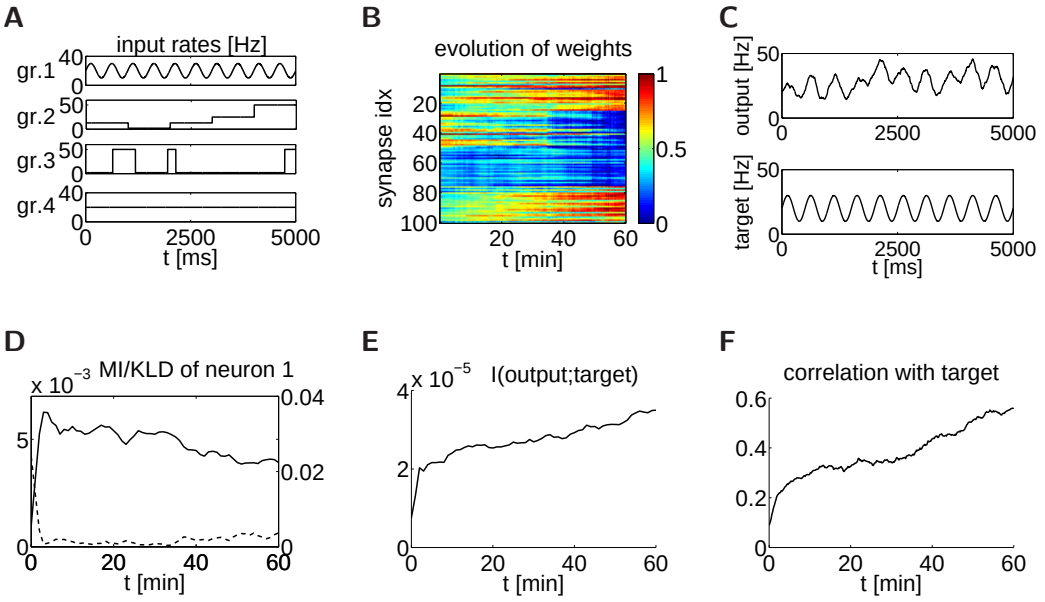

Figure 2.6: **Extracting a single rate modulation with the spike-based rule** (2.18)**.** (**A**) Modulation of input rates for each of the four groups. (**B**) Evolution of weights during 60 minutes of learning (red: strong synapses, $w_{ij} \approx 1$, blue: depressed synapses, $w_{ij} \approx 0$.) Weights were initialized randomly between 0.10 and 0.12, $\alpha = 5 \cdot 10^{-4}$, $\beta = 10^3$, $\gamma = 10$. Each group receives Poisson input with a different rate modulation $r_i(t)$; the rate modulation of the target signal is the same as for input group 1, $r_T(t) = r_1(t)$. (**C**) Output rate and rate of the target signal during 5 seconds after learning. (**D**) Evolution of the average mutual information per time bin (solid line, left scale) between input and output, and the Kullback-Leibler divergence per time bin (dashed line, right scale) as a function of time. Averages are calculated over segments of 1 minute. (**E**) Evolution of the average mutual information per time bin between output and the target signal as a function of time. (**F**) Trace of the time-varying correlation between output rate and rate of the target signal during learning. Correlation coefficients are calculated every 10 seconds.

interesting, we change the rate of the target signal to the mean of rates of group 1 and 3, $r_T(t) = (r_1(t) + r_3(t))/2$, after 15 minutes . Furthermore, to investigate the effect of removing the target signal after some time, we switch it off after 45 minutes ($r_T(t) = 0$).

Figure 2.7 shows the performance of the simplified learning rule (2.21) for this task. In the panel 2.7B we see that weights grow initially for input groups 1 and 2 and remain depressed for the other inputs, as expected. After 15 minutes, as the firing rate combination of the target signal changes, the weights of group 2 are weakened whereas the efficacies of the third group now start to grow. This means that the learning rule is able to adapt to new situations where the relevant target signal changes. However, the final distribution of synaptic efficacies persists when the target signal is removed after 45 minutes.

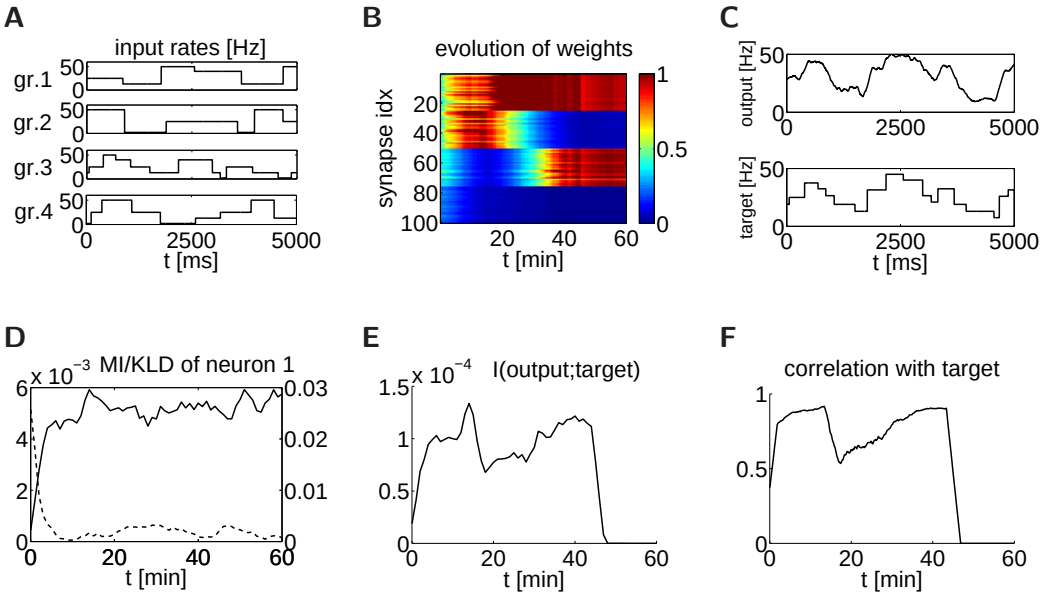

Figure 2.7: **Extracting input components that are indirectly and just during certain time points related to the target signal with the rate-based rule** (2.21)**.** (**A**) Modulation of input rates for each of the four groups. Each group i receives Poisson input with a different rate modulation $r_i(t)$. (**B**) Evolution of weights during 60 minutes of learning (red: strong synapses, $w_{ij} \approx 1$, blue: depressed synapses, $w_{ij} \approx 0$.) Weights were initialized randomly between 0.10 and 0.12, $\alpha = 10^{-3}$, $\beta = 5 \cdot 10^3$, $\gamma = 10$. Initially, $r_T(t) = 1/2(r_1(t) + r_2(t))$; after 15 minutes it changes to $r_T(t) = 1/2(r_1(t) + r_3(t))$. After 45 minutes, $r_T(t) = 0$. (**C**) Output rate and rate of the target signal during 5 seconds just before the target signal is removed. (**D**) Evolution of the average mutual information between input and output per time bin (solid line, left scale), and the Kullback-Leibler divergence per time bin (dashed line, right scale) as a function of time. Averages are calculated over segments of 1 minute. (**E**) Evolution of the average mutual information per time bin between output and the target signal as a function of time. (**F**) Trace of the correlation between output rate and rate of the target signal during learning. Note that the target signal is changed after 15 minutes, and set to 0 after 45 minutes. Correlation coefficients are calculated every 10 seconds.

2.5.3 Extracting spike-spike correlations

So far we have only considered rate coding, i.e., the information was encoded in the firing rates of the spike trains. But can the proposed learning rule also take into account information that is contained in the spike timings rather than in the firing rates? In the next experiment we investigate the effect of spike-spike correlations between the target spike train and parts of the input for the spike-based learning rule (2.18). All input spike trains and the target spike train are now generated by a Poisson process at a constant rate of 20Hz. However, different correlation groups are established within the inputs in the following way: The first 25 inputs are strongly correlated with the target spike train (with a coefficient of 0.5), the second 25 synapses have weaker correlations with the target spike train (coefficient

0.2). The remaining 50 inputs are uncorrelated with the target spike train, however, inputs 51 to 75 are pairwise correlated with a coefficient of 0.5, and inputs 76 to 100 are uncorrelated. Inputs belonging to different groups are also uncorrelated. Correlated spike trains are generated by the procedure described in (Gütig et al., 2003; Legenstein et al., 2005).

Figure 2.8 shows that strong weights grow for those synapses where the input has spike-spike correlations with the target spike train. Because the first group of inputs is correlated more strongly than the second group, weights from the first group reach their maximum value of 1 whereas those for the second group only grow up to a value of about 0.5. That is, the learning rule is sensitive to different levels of correlation. Note that the information conveyed by spike-spike correlations is about one order of magnitude larger than in the previous experiments with rate coding. In Figure 2.8D the correlation between the output and the target spike train is bounded from above by the maximum correlation of inputs with the target spike train (0.5).

2.5.4 Extracting information that is relevant for two different target signals

We use a setup as in Figure 2.1A where we want to maximize the information which the output Y_1^K of a learning neuron conveys about two target signals, Y_2^K and Y_3^K. If the target signals are statistically independent from each other we can optimize the mutual information to each target signal separately, i.e., we include the term $\beta(I(\mathbf{Y}_1^K;\mathbf{Y}_2^K)+I(\mathbf{Y}_1^K;\mathbf{Y}_3^K))$ in the objective function (2.11). This leads to an update rule

$$\frac{\Delta w_{1j}^k}{\Delta t} = -\alpha C_{1j}^k \left[B_1^k(-\gamma) - \beta \Delta t \left(B_{12}^k + B_{13}^k \right) \right], \tag{2.30}$$

where B_{12}^k and B_{13}^k are the postsynaptic terms (2.20) sensitive to the statistical dependence between the output and target signals 1 and 2, respectively.

In this experiment we demonstrate that it is possible to consider two very different kinds of target signals: one target spike train has a similar rate modulation as one part of the input, while the other target spike train has a high spike-spike correlation with another part of the input. The first two of the four input groups consist of rate modulated Poisson spike trains, where the rate of the first 25 inputs is modulated by a Gaussian white-noise signal with mean 20Hz that has been low-pass filtered with a cut-off frequency of 5Hz. Synapses 26 to 50 receive the burst signal described in section 2.5.1, which was used there for input group 3 (see Figure 2.9A). Spike trains from the remaining groups 3 and 4 are Poisson spike trains at a constant rate of 20Hz, but have spike-spike correlations with a coefficient of 0.5 within each group. However, spike trains from different groups are uncorrelated. The first target spike train is chosen to have a similar rate modulation as the inputs from group 1; Gaussian random noise is superimposed on the rate with a standard deviation of 2Hz. The second target spike train is correlated with inputs from group 3 (with a coefficient of 0.5), but uncorrelated to inputs from group 4. Furthermore, both target signals are silent during random intervals: at each time step, the rate

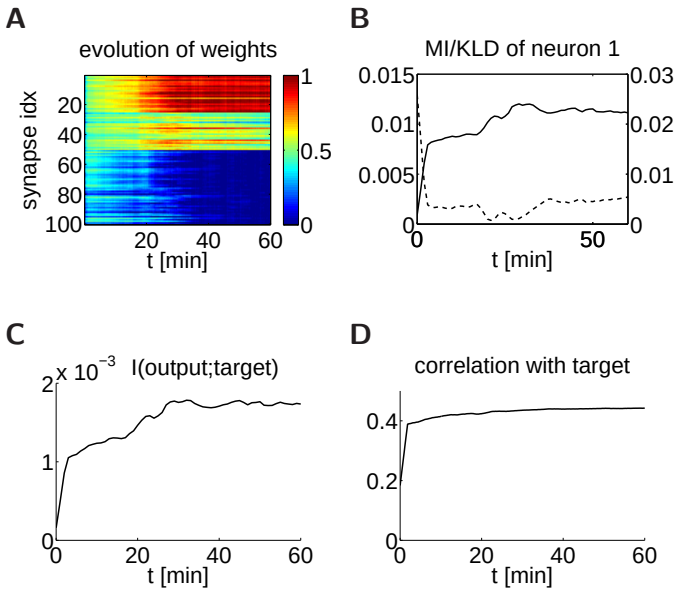

Figure 2.8: **Extracting spike-spike correlations with the spike-based learning rule (2.18).** (**A**) Evolution of weights during 60 minutes of learning (red: strong synapses, $w_{ij} \approx 1$, blue: depressed synapses, $w_{ij} \approx 0$.) Weights were initialized randomly between 0.10 and 0.12, $\alpha = 10^{-4}$, $\beta = 10^2$, $\gamma = 50$. All inputs and the target spike train are Poisson spike trains at a constant rate of 20Hz. Input group 1 and the target spike train are correlated with a coefficient of 0.5, between input group 2 and the target spike train a correlation coefficient of 0.2 is established. Group 3 is also correlated with 0.5, but uncorrelated to the target spike train, and group 4 is uncorrelated at all. Spike trains from different groups are uncorrelated. (**B**) Evolution of the average mutual information per time bin (solid line, left scale) between input and output, and the Kullback-Leibler divergence per time bin (dashed line, right scale) as a function of time. Averages are calculated over segments of 1 minute. (**C**) Evolution of the average mutual information per time bin between output and the target signal as a function of time. (**D**) Trace of the current spike-spike correlation between the output spike train and the target spike train during learning. Correlation coefficients are calculated every 10 seconds. This experiment shows that the neuron learns with the IB learning rule to extract information from high dimensional input streams that is contained in the spike times.

of each target signal is independently set to 0 with a certain probability (10^{-5}) and remains silent for a duration chosen from a Gaussian distribution with mean 5s and SD 1s (minimum duration is 1s). Hence this experiment tests whether learning works even if the target signals are not available all of the time.

Figure 2.9 shows that strong weights evolve for the first and third group of synapses, whereas the efficacies for the remaining inputs are depressed. Both groups with growing weights are correlated with one of the target signals, therefore the mutual information between output and target spike trains increases. Since spike-spike correlations convey more information than rate modulations synaptic efficacies develop more strongly to group 3 (the group with spike-spike correlations). This results in an initial decrease in correlation with the rate-modulated target signal to

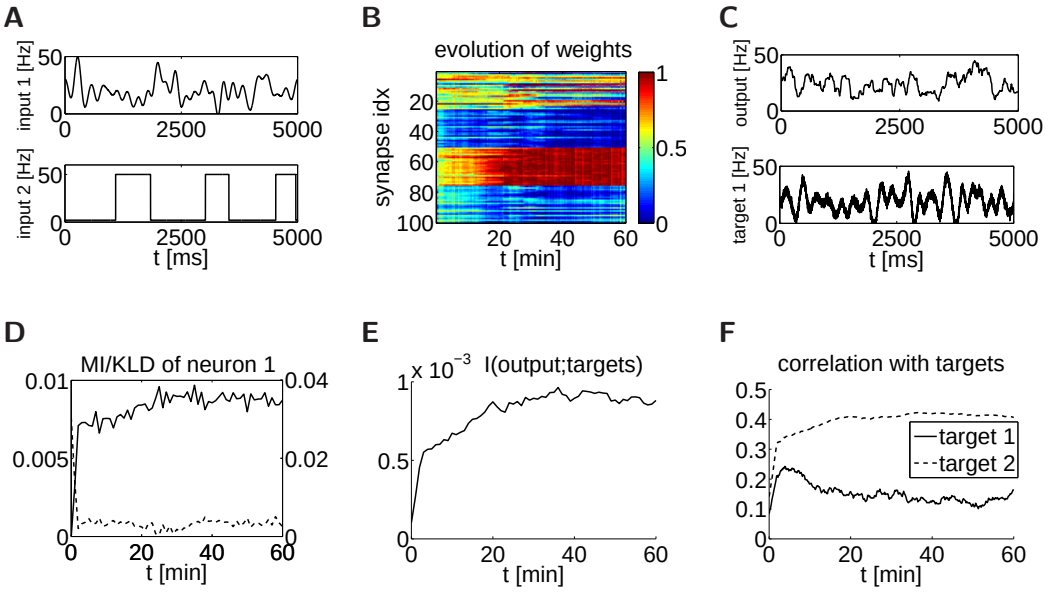

Figure 2.9: **Using two target signals at the same time with the spike-based rule (2.18).** (**A**) Modulation of input rates to input groups 1 and 2. (**B**) Evolution of weights during 60 minutes of learning (red: strong synapses, $w_{ij} \approx 1$, blue: depressed synapses, $w_{ij} \approx 0$.) Weights were initialized randomly between 0.10 and 0.12, $\alpha = 10^{-4}$, $\beta = 2 \cdot 10^3$, $\gamma = 50$. Input groups 1 and 2 receive Poisson spike trains with different rate modulations; groups 3 and 4 receive constant rate Poisson at 20Hz, but each group is correlated with a coefficient of 0.5, and spike trains from different groups are uncorrelated. The first target spike train is a Poisson spike train with the same rate modulation as group 1, superimposed with Gaussian noise ($\sigma = 2$Hz). The second target spike train has a constant rate of 20Hz and is correlated with coefficient 0.5 to input group 3. (**C**) Output rate and rate of target signal 1 during 5 seconds after learning. (**D**) Evolution of the average mutual information per time bin (solid line, left scale) between input and output and the Kullback-Leibler divergence per time bin (dashed line, right scale) as a function of time. Averages are calculated over segments of 1 minute. (**E**) Evolution of the average mutual information per time bin between output and both target spike trains as a function of time. (**F**) Trace of the current correlation between output rate and rate of target signal 1 (solid line) and the spike-spike correlation (dashed line) between the output and target spike train 2 during learning. Correlation coefficients are calculated every 10 seconds.

the benefit of higher correlation with the second target spike train. However, after about 30 minutes when the weights become stable, the correlations as well as the mutual information quantities stay roughly constant.

2.5.5 Extracting information that is uncorrelated with the target signal, but has higher order statistical dependencies

So far we have analyzed the information bottleneck setup only for situations where the target signal is correlated to parts of the input (either via rate correlations or spike-spike correlations). To show that the learning rule is also able to extract

statistical dependencies of higher order, we try to extract uncorrelated, but still statistically dependent information. In this experiment, we use again rate coding and choose the firing rate of the target signal to be a function of one of the input rate modulations as to induce strong statistical dependence between the target spike train and this input group. In order to decorrelate the target signal from this input, a whitening transformation is applied (see Appendix B.2).

We generate the rate modulations for the four input groups $r_1(t), \ldots, r_4(t)$ in the same way as in the experiment described in section 2.5.3, i.e., piecewise constant rates chosen randomly out of the set $\{2\text{Hz}, 13\text{Hz}, 25\text{Hz}, 40\text{Hz}, 50\text{Hz}\}$, and the duration during which the rate is constant is drawn uniformly from the interval $[0\text{s}, 1\text{s}]$. Inputs from the same group share the same rate modulation, inputs from different groups are statistically independent, since the rates are drawn independently for each group. The rate of the target signal is chosen to be a function of the first input rate, i.e., $r_T(t) = f(r_1(t))$, where $f(2) = 13\text{Hz}$, $f(13) = 25\text{Hz}$, $f(25) = 40\text{Hz}$, $f(40) = 50\text{Hz}$, and $f(50) = 2\text{Hz}$. In this way, statistical dependence has been established between the first input group and the target spike train. Now, the whitening transformation is applied to decorrelate the rate modulation of the first input group, $r_1(t)$, and the target signal, $r_T(t)$, yielding $\tilde{r}_1(t)$ and $\tilde{r}_T(t)$. Finally, the input spike trains are generated by inhomogeneous Poisson processes with the rates $\tilde{r}_1(t)$, $r_2(t)$, $r_3(t)$, and $r_4(t)$, and the target spike train is drawn from $\tilde{r}_T(t)$. For this experiment, we choose $\tilde{g} = 20\text{Hz}$.

The performance of the rate-based rule on this task is shown in Figure 2.10. It can be seen that weights reach values close to maximal efficacy for the statistically dependent group (group 1) and finally get depressed for the remaining inputs. The output is now uncorrelated, but still statistically dependent to the target signal. Note that the mutual information between output and target signal increases whereas the correlation stays around 0. This means that the learning rule is also sensitive to higher order statistical dependencies.

2.6 Extracting independent components

With a slight modification in the objective function (2.11) the learning rule allows us to extract statistically independent components from an ensemble of input spike trains. We consider two neurons receiving the same input at their synapses (see Figure 2.1B). For both neurons $i = 1, 2$ we maximize information transmission under the constraint that their outputs stay as statistically independent from each other as possible. That is, we maximize

$$\tilde{L}_i = I(\mathbf{X}^K; \mathbf{Y}_i^K) - \beta I(\mathbf{Y}_1^K; \mathbf{Y}_2^K) - \gamma D_{KL}(P(Y_i^K) || \tilde{P}(Y_i^K)). \tag{2.31}$$

Since the same terms (up to the sign) are optimized in (2.11) and (2.31) we can derive a gradient ascent rule for the weights of neuron i, w_{ij}, analogously to section 2.3:

$$\frac{\Delta w_{ij}^k}{\Delta t} = \alpha C_{ij}^k \left[B_i^k(\gamma) - \beta \Delta t B_{12}^k \right] \tag{2.32}$$

(see Table 2.1 for a definition of the terms in this equation).

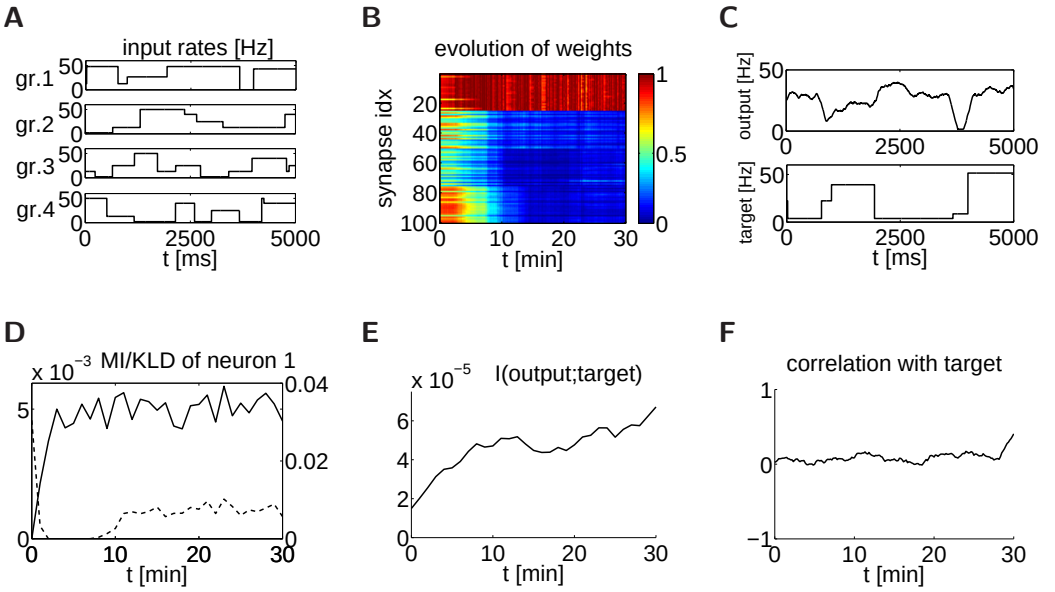

Figure 2.10: **Extracting uncorrelated but statistically dependent information with the rate-based rule** (2.21)**.** (**A**) Modulation of input rates to input groups 1 to 4. (**B**) Evolution of weights during 30 minutes of learning (red: strong synapses, $w_{ij} \approx 1$, blue: depressed synapses, $w_{ij} \approx 0$.) Weights were initialized randomly between 0.10 and 0.12, $\alpha = 10^{-6}$, $\beta = 2 \cdot 10^7$, $\gamma = 50$. Each input group receives Poisson input with different rate modulations, the rate modulation of the target is a function of the rate of input group 1. The rate of the target signal and the rate of input group 1 are decorrelated using the whitening transform described in the text. Nevertheless the learning rule picks out these inputs that have statistical dependencies with the target signal. (**C**) Output rate and rate of the target signal during 5 seconds after learning. (**D**) Evolution of the average mutual information per time bin (solid line, left scale) between input and output and the Kullback-Leibler divergence per time bin (dashed line, right scale) as a function of time. Averages are calculated over segments of 1 minute. (**E**) Evolution of the average mutual information per time bin between output and target spike train as a function of time. One clearly sees that this mutual information keeps increasing, whereas the mutual information between input and output (see **D**) stays on the same level. (**F**) Trace of the correlation between output rate and rate of the target signal during learning. Correlation coefficients are calculated every 10 seconds.

In order to compare this rule with the BCM model as in section 2.4.3 we consider the weight change of neuron 1 for the rate-based rule derived for the simplified neuron model,

$$\frac{\Delta w_{1j}^k}{\Delta t} = \alpha \nu_j^{pre,k} \tilde{\Phi}(\nu_1^k, \nu_2^k), \tag{2.33}$$

where $\tilde{\Phi}(\nu_1^k, \nu_2^k)$ is given by

$$\tilde{\Phi}(\nu_1^k, \nu_2^k) = f(\nu_1^k) \left\{ \log \left[\frac{\nu_1^k}{\bar{\nu}_1^k} \left(\frac{\bar{\nu}_1^k}{\tilde{g}} \right)^{-\gamma} \right] - \beta \Delta t \left[\nu_2^k \log \phi - \bar{\nu}_2^k (\phi - 1) \right] \right\}, \tag{2.34}$$

with $\phi = \bar{\nu}_{12}^k/(\bar{\nu}_1^k\bar{\nu}_2^k)$.

Compared to the IB rule (2.24), the sign of the weight update has changed in (2.33), reflecting the different signs in the first two terms of the objective function (2.31) as compared to (2.11). The synaptic modification function (2.34) is the same as $\Phi(\nu_1^k, \nu_2^k)$ in equation (2.25), except that γ in (2.25) is replaced by $-\gamma$. In the following discussion, we consider the case where the output rate of neuron 1 is already close to the target firing rate, so that $\bar{\nu}_1^k \approx \tilde{g}$. In this case, $\tilde{\Phi}(\nu_1^k, \nu_2^k)$ is approximately equal to $\Phi(\nu_1^k, \nu_2^k)$ and Figure 2.5 qualitatively also applies for $\tilde{\Phi}$.

Analogous arguments as in section 2.4.3 can be applied when comparing this rule with the BCM model. Because of the Hebbian nature of (2.33) values of Φ above 0 produce LTP and values below 0 produce LTD (see Figure 2.5). Again, for the special case $\phi = 1$ (see Figure 2.5B) the outputs are uncorrelated and the learning rule reduces to the classical BCM rule, i.e., the output of neuron 2, ν_2^k, has no influence on the weight change Δw_{1j}^k of neuron 1. In case of anti-correlated outputs of the two neurons ($\phi < 1$, see Figure 2.5A) the learning rule will try to make them more correlated by increasing ν_1^k for large ν_2^k and decreasing ν_1^k for small ν_2^k. On the other hand, if the outputs are correlated ($\phi > 1$, see Figures 2.5C and 2.5D), anti-correlations will be increased: For large values of ν_2^k the output firing rate ν_1^k tends to decrease; for small values of ν_2^k it increases. In this way, the learning rule tries to make these outputs statistically independent. Again, note that correlation and anti-correlation both contribute in the same way to mutual information.

2.6.1 An approximation of the learning rule

The term B_{12}^k (2.20) in the learning rule (2.32) is nonlocal and difficult to implement by a spiking neuron in reality. In the following we provide an approximation to the learning rule (2.32) in which we implement the effect of the term B_{12}^k by modifying the value $g(u_i(t^k))$ in the term B_i^k. This could provide an idea how this learning rule might possibly be implemented in a biologically realistic circuit of neurons. More precisely, we let the weights of neuron i evolve according to the learning rule

$$\frac{\Delta w_{ij}^k}{\Delta t} = \alpha C_{ij}^k \hat{B}_i^k(\gamma), \tag{2.35}$$

which is similar to the generalized BCM rule for spiking neurons presented in section 2.2, where

$$\begin{aligned}\hat{B}_1^k(\gamma) = \frac{y_1^k}{\Delta t} \log\left[\frac{\hat{g}_1(t^k)}{\bar{g}_1(t^k)}\left(\frac{\tilde{g}}{\bar{g}_1(t^k)}\right)^\gamma\right] \\ - (1-y_1^k) R_1(t^k)\left[\hat{g}_1(t^k) - (1+\gamma)\bar{g}_1(t^k) + \gamma\tilde{g}\right],\end{aligned} \tag{2.36}$$

is $B_i^k(\gamma)$ with a modified gain function $\hat{g}_i(t^k)$ (see Table 2.3). Note that we do not change the actual gain function (i.e., firing behavior) of the neuron, the modified gain function $\hat{g}_i(t^k)$ is only effective in the learning rule.

To find the desired expression for $\hat{g}_i(t^k)$, we compare the combined postsynaptic term $B_i^k(\gamma) - \beta\Delta t B_{12}^k$ in (2.32) with the simple postsynaptic term $\hat{B}_i^k(\gamma)$ (2.36) for

both neurons and for the two cases that the neuron itself or the other neuron has emitted a spike (see Appendix B.3). This results in a modified gain function for the learning rule of neuron $i = 1, 2$ of

$$\hat{g}_i(t^k) = g(u_i(t^k)) \cdot a_i(t^k)^{y_i^k(1-y_{3-i}^k)} + b_i(t^k) y_{3-i}^k (1 - y_i^k). \tag{2.37}$$

The term

$$a_i(t^k) = \exp\left[R_{3-i}(t^k)\beta\Delta t \left(\frac{\bar{g}_{12}(t^k)}{\bar{g}_i(t^k)} - \bar{g}_{3-i}(t^k)\right)\right] \tag{2.38}$$

corresponds to a multiplicative change of $g(u_i(t^k))$ in case of spikes of neuron i itself. If the outputs have been correlated (i.e., $\bar{g}_{12}(t^k) > \bar{g}_1(t^k)\bar{g}_2(t^k)$) the modified gain in (2.35) is increased, if the outputs have been anti-correlated ($\bar{g}_{12}(t^k) < \bar{g}_1(t^k)\bar{g}_2(t^k)$) it is decreased. If on the other hand a spike is elicited by the other neuron (neuron $3 - i$) the value $g(u_i(t^k))$ is modified additively by the term

$$b_i(t^k) = -\beta\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_{3-i}(t^k)} - \bar{g}_i(t^k)\right]. \tag{2.39}$$

In case of correlated outputs it is decreased, in case of anti-correlated outputs it is increased.

Note that equations (2.35) to (2.39) only provide an approximation to the learning rule (2.32) because we have considered only the cases where one of the two neurons spikes. The approximation presented here is still not local because the modified value $\hat{g}_i(t^k)$ still depends on nonlocal variables, e.g., the average product of firing rates $\bar{g}_{12}(t^k)$. However, it indicates what a real biological learning rule would have to approximate. Each neuron needs information about the firing behavior of both neurons. In particular, a circuit of interneurons would be necessary to implement some of the terms in equations (2.37) to (2.39).

2.6.2 Extracting different correlation groups

Figure 2.12 shows the results of an experiment where two neurons receive the same Poisson input with a rate of 20Hz at their 100 synapses. The input is divided into two groups of 40 spike trains each, such that synapses 1 to 40 and 41 to 80 receive correlated input with a correlation coefficient of 0.5 within each group, however, any spike trains belonging to different input groups are uncorrelated. The remaining 20 synapses receive uncorrelated Poisson input (see Figure 2.11 for a sample of such input spike trains). Weights close to the maximal efficacy $w_{max} = 1$ are developed for one of the groups of synapses that receives correlated input (group 2 in this case) whereas those for the other correlated group (group 1) as well as those for the uncorrelated group (group 3) stay low. Neuron 2 develops strong weights to the other correlated group of synapses (group 1) whereas the efficacies of the second correlated group (group 2) remain depressed, thereby trying to produce a statistically independent output. For both neurons the mutual information is maximized and the target output distribution of a constant firing rate of 30Hz is approached well. After an initial increase in both the mutual information and in the correlation between the outputs, where the weights of both neurons start to

Approximation of the learning rule for extracting independent components:

The weights w_{ij} of neuron $i = 1, 2$ evolve according to the generalized BCM rule for spiking neurons. The weight change Δw_{ij}^k at time $t^k = k\Delta t$ is given by

$$\frac{\Delta w_{ij}^k}{\Delta t} = \alpha C_{ij}^k B_i^k(\gamma) \tag{2.35}$$

with a learning rate $\alpha > 0$ and optimization parameter $\gamma > 0$.

The correlation term C_{ij}^k and the postsynaptic term $B_i^k(\gamma)$ are given by:

$$C_{ij}^k = C_{ij}^{k-1}\left(1 - \frac{\Delta t}{\tau_C}\right) + \sum_{n=1}^{k} \varepsilon(t^k - t^n) x_j^n \frac{g'(u_i(t^k))}{g(u_i(t^k))}\left[y_i^k - \rho_i^k\right] \tag{2.16}$$

$$\begin{aligned} B_i^k(\gamma) = {} & \frac{y_i^k}{\Delta t}\log\left[\frac{\hat{g}_i(t^k)}{\bar{g}_i(t^k)}\left(\frac{\tilde{g}}{\bar{g}_i(t^k)}\right)^{\gamma}\right] \\ & - (1 - y_i^k) R_i(t^k)\left[\hat{g}_i(t^k) - (1+\gamma)\bar{g}_i(t^k) + \gamma\tilde{g}\right] \end{aligned} \tag{2.36}$$

(compare to (2.19) in Table 2.1).
The original gain value $g(u_i(t^k))$ is modified both additively and multiplicatively:

$$\hat{g}_i(t^k) = g(u_i(t^k)) \cdot a_i(t^k)^{y_i^k(1-y_{3-i}^k)} + b_i(t^k) y_{3-i}^k (1 - y_i^k), \tag{2.37}$$

where $y_i^k \in \{0, 1\}$ indicates an output spike of neuron i at time t^k.

If neuron i itself has spiked the value $g(u_i(t^k))$ is multiplied with the following factor:

$$a_i(t^k) = \exp\left[R_{3-i}(t^k)\beta\Delta t\left(\frac{\bar{g}_{12}(t^k)}{\bar{g}_i(t^k)} - \bar{g}_{3-i}(t^k)\right)\right] \tag{2.38}$$

If the other neuron (neuron $3 - i$) has spiked the following term is added to the value $g(u_i(t^k))$:

$$b_i(t^k) = -\beta\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_{3-i}(t^k)} - \bar{g}_i(t^k)\right] \tag{2.39}$$

Table 2.3: **Summary of the approximation of the learning rule for extracting independent components.**

grow simultaneously, these amounts drop as both neurons develop strong efficacies to different parts of the input.

2.6.3 Comparison with other neural ICA learning rules

Neural learning algorithms based on information optimization principles, such as independent component analysis (ICA) (Hyvärinen et al., 2001), have previously been derived for rate-based models (Hyvärinen and Oja, 1996, 1998). However, an application to spiking neurons has still been missing. In this section we have presented an ICA rule for spiking neurons which is not only able to detect statistical

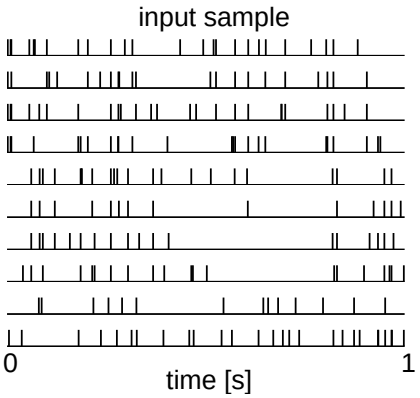

Figure 2.11: **Demonstration of the difficulty of the ICA task for spike trains.** Shown are 10 spike trains for 1s which represent 10 out of 100 inputs in the experiment described in Figure 2.12. The top 4 spike trains (group 1) are correlated with a correlation coefficient of 0.5, as are spike trains 5-8 (group 2). However, spike trains from different groups are uncorrelated. The remaining bottom two input spike trains (group 3) are uncorrelated. Obviously it is quite difficult to detect which spike trains are correlated, due to their rather weak correlation.

dependencies between the input rates, but also between the timing of individual spikes, as shown in the experiment in Figure 2.12. Furthermore, while in ICA one usually assumes that the data is generated by a linear combination of statistically independent sources, we do not assume any model on how the data is generated. The experiment in Figure 2.12 also shows that our learning rule performs blind source separation even if the sources are not linearly mixed (which is not possible for a spiking input where the information is encoded in spike timings).

2.7 Discussion

Information bottleneck (IB) and Independent component analysis (ICA) have been proposed as principles for unsupervised learning in lower cortical areas, however, learning rules that can implement these principles with spiking neurons have been missing. So far, synaptic update rules optimizing information-theoretic objectives have been presented mainly for rate models and real-valued units, e.g., (Linsker, 1989; Bell and Sejnowski, 1995; Becker, 1996). In this Chapter we have derived from information-theoretic principles learning rules which enable a stochastically spiking neuron to solve these tasks. We have shown in section 2.4.3 that these rules can be viewed as an extension to the classical Bienenstock-Cooper-Munro (BCM) rule (Bienenstock et al., 1982) and to its generalized variant for spiking neurons (Toyoizumi et al., 2005). Furthermore, we have demonstrated how they are related to traditional Information bottleneck algorithms (see section 2.4.4) and neural ICA learning rules (see section 2.6.3). Our learning rules, which are optimal from the perspective of information theory, are not local in the sense that they use only information that is available at a single synapse without an auxiliary network of

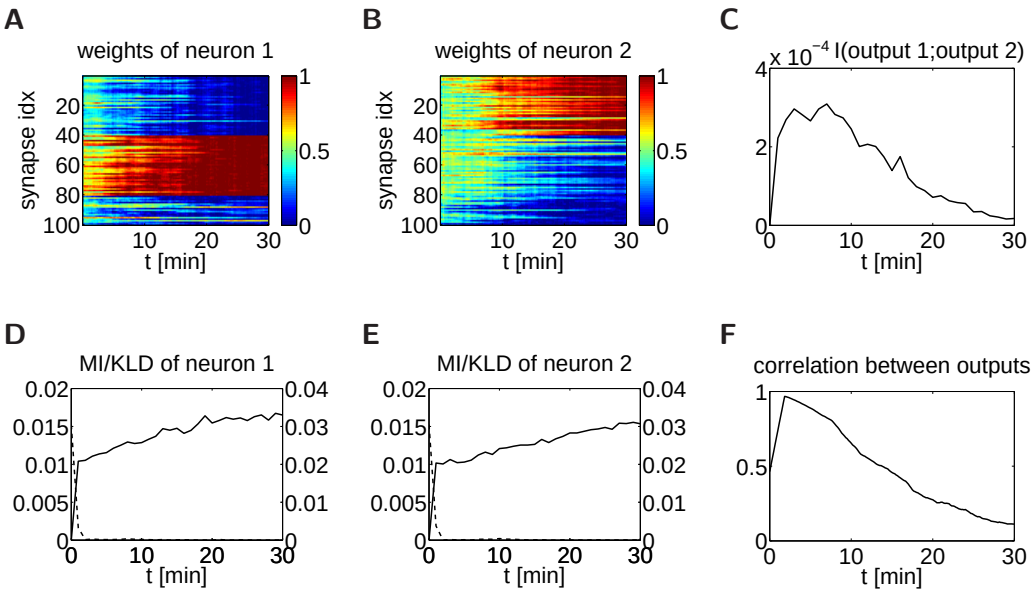

Figure 2.12: **Extracting independent components from 100 input spike trains.** (**A, B**) Evolution of weights during 30 minutes of learning for both postsynaptic neurons (red: strong synapses, $w_{ij} \approx 1$, blue: depressed synapses, $w_{ij} \approx 0$.) Weights were initialized randomly between 0.10 and 0.12, $\alpha = 5 \cdot 10^{-4}$, $\beta = 100$, $\gamma = 50$. (**C**) Evolution of the average mutual information per time bin between both output spike trains as a function of time. (**D, E**) Evolution of the average mutual information per time bin (solid line, left scale) between input and output and the Kullback-Leibler divergence per time bin for both neurons (dashed line, right scale) as a function of time. Averages are calculated over segments of 1 minute. (**F**) Trace of the current correlation between both output spike trains during learning. Correlation coefficients are calculated every 10 seconds.

interneurons or other biological processes. But they tell us what type of information would have to be ideally provided by such auxiliary network, and how the synapse should change its efficacy in order to approximate a theoretically optimal learning rule.

The learning rule for ICA that we have derived appears to be the first ICA learning rule for spiking neurons. We have demonstrated in Figures 2.11 and 2.12 that in particular this learning rule enables spiking neurons to discover and remove dependencies in their input spike trains that are not encoded through correlations or other dependencies between their firing rates, but through correlations between the timing of individual spikes. But this ICA rule is also able to remove dependencies in firing rates.

Information bottleneck optimization is another and potentially more powerful method for deriving rules for learning that might shape the output of projection neurons which send selected information to higher cortical areas, or downwards to the thalamus. In contrast to ICA, IB optimization need not be driven exclusively by the statistics of sensory input signals. Rather, IB optimization allows to select that information from sensory inputs that is related to inputs from another sensory

modality, to proprioceptive feedback, to expectations, or to rewards. Hence, it may contribute to the emergence of synergistic internal codes for relevant parts of the external world, which combine information from different sensory modalities (see (Calvert et al., 2004)), causing in particular effects such as improved understanding of spoken language if the face of the speaker can be observed, and to goal-oriented and task-dependent sensory processing (Sigala and Logothetis, 2002; Shuler and Bear, 2006; Fritz et al., 2003). Hence, IB learning rules share aspects both of unsupervised and supervised[5] learning. We have demonstrated through five computer experiments that the IB-learning rules for spiking neurons that we have derived are capable to extract information simultaneously from rates and from spike trains (see Figures 2.6, 2.8, and 2.9), to extract input signals that are only partially related to the target signal (since the target is a sum of several input signals, see Figure 2.7), and to extract information that is related to two simultaneously presented target signals (which encode information in two different ways, see Figure 2.9). We have also demonstrated in Figure 2.10 that the learning rule can learn to extract information from the input that is not correlated with the target signal, but is related through higher order statistical dependencies. Finally we have demonstrated that the learning rules that we have derived work quite fast, in most cases within a few minutes. We have also demonstrated that they are very stable (hence do not require any regulation of learning rates), since their performance does not degrade during experiments of long duration. Furthermore, the firing rate of the learning neurons always stays within the desired range. In future work it would be interesting to investigate also applications of these learning rules to signal processing problems (e.g., noise filtering), since the IB approach promises to provide optimal solutions to some of these tasks.

The results of this Chapter only show that biological neurons could in principle carry out ICA and Information bottleneck analysis, and we have shown how close-to-optimal learning rules for spiking neurons would look like. We also have argued that both learning principles are very useful for any multi-sensory distributed cognitive system. This poses the challenge to neurophysiology to test through experiments in vivo and in vitro to what extent (and where) these learning principles are implemented in neural systems, and how they are implemented through synaptic plasticity.

2.8 Acknowledgements

This Chapter is based on the paper *Spiking neurons can learn to solve information bottleneck problems and to extract independent components* by myself (SK), Robert Legenstein (RL), and my supervisor Wolfgang Maass (WM). The derivation of the spike based learning rule and its analysis was performed by SK. The simplified rate

[5]Note that maximizing the mutual information between the output of a neuron and a target signal offers an interesting alternative to supervised learning for neurons, where the "code" that the neuron uses is left unspecified. While in a supervised learning task the target output is prescribed, and should be reproduced as exactly as possible by the learning unit, this is not the case for an Infomax problem. Rather, Information bottleneck can be viewed as a supervised selection of what is *relevant*, while the learning process itself and the choice of neural codes is unsupervised.

based rule was developed by RL and analyzed by both RL and SK. The simulation experiments were designed and conducted by SK. SK wrote the paper, with additional input from WM and RL. The paper benefitted from helpful discussions with Wulfram Gerstner and Jean-Pascal Pfister and was written under partial support by the Austrian Science Fund FWF, # S9102-N13 and # P17229-N04, and was also supported by PASCAL, project # IST2002-506778, and FACETS, project # 15879, of the European Union.

Chapter 3

A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction

Contents

In this Chapter it is shown that a particular unsupervised learning algorithm based on the temporal slowness principle, slow feature analysis, is able to approximate the classification capability of a well-known supervised learning method, Fisher's linear discriminant. Furthermore, the capability of this learning method as a possible readout mechanism of a generic cortical microcircuit model is analyzed. This Chapter is based on the paper A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction *by Stefan Klampfl and Wolfgang Maass (*Neural Computation*, 22(12):2979-3035, 2010).*

Neurons in the brain are able to detect and discriminate salient spatio-temporal patterns in the firing activity of presynaptic neurons. It is open how they can learn to achieve this, especially without the help of a supervisor. We show that a well-known unsupervised learning algorithm for linear neurons, slow feature analysis (SFA), is able to acquire the discrimination capability of one of the best algorithms for supervised linear discrimination learning, the Fisher linear discriminant (FLD), given suitable input statistics. We demonstrate the power of this principle by showing that it enables readout neurons from simulated cortical microcircuits to learn without any supervision to discriminate between spoken digits, and to detect repeated firing patterns that are embedded into a stream of noise spike trains with the same firing statistics. Both these computer simulations and our theoretical

analysis show that slow feature extraction enables neurons to extract and collect information that is spread out over a trajectory of firing states that lasts several hundred ms. In addition, it enables neurons to learn without supervision to keep track of time (relative to a stimulus onset, or the initiation of a motor response). Hence these results elucidate how the brain could compute with trajectories of firing states, rather than only with fixed point attractors. It also provides a theoretical basis for understanding recent experimental results on the emergence of view- and position-invariant classification of visual objects in inferior temporal cortex.

3.1 Introduction

The brain is able to extract an astonishing amount of information from its environment without a supervisor or teacher that tells the brain how an external stimulus should be classified. Experimental data show that one method which the brain uses in order to learn the categorization of external objects without a supervisor is the temporal slowness learning principle, which exploits the fact that temporally adjacent sensory stimuli are likely to be caused by the same external object. More precisely, experimental results from the lab of DiCarlo (Cox et al., 2005; Li and DiCarlo, 2008) (see DiCarlo and Cox, 2007, for a review) show that this simple heuristic is sufficient for the formation of position- and view-invariant representations of visual objects in higher cortical areas. This was tested in clever experiments by altering the probability that different objects caused temporally adjacent firing states in primary visual cortex (the external visual stimuli were swapped during the transient blindness while a saccade was performed). Human subjects were reported to merge different visual objects – presented at different retina locations – into single visual percepts as a result of this manipulation of the temporal statistics of visual inputs. Also the firing response of neurons in monkey area IT was reported to change accordingly. As a result of these data it was hypothesized in (Li and DiCarlo, 2008) that "unsupervised temporal slowness learning may reflect the mechanism by which the visual stream builds and maintains tolerant object representations". But a rigorous theoretical basis for the emergent discrimination capability of this unsupervised temporal slowness learning principle proposed by (Li and DiCarlo, 2008) has been missing. Such theoretical foundation, which relates the statistics of the sequence of external stimuli to the emergent discrimination capability of this unsupervised learning method, is provided in this Chapter.

There have been a number of approaches to learn invariant representations in an unsupervised manner from the contingency of temporally adjacent inputs, i.e., by extracting features that vary on a slow time scale (e.g., Földiak, 1991; Mitchison, 1991; Becker and Hinton, 1992; Stone and Bray, 1995). We focus on one particularly transparent computational mechanism for unsupervised temporal slowness learning: slow feature analysis (SFA), introduced by (Wiskott, 1998; Wiskott and Sejnowski, 2002). SFA transforms a (usually high-dimensional) time series $\mathbf{x}$ into an output y, and minimizes the temporal variation of y under the additional constraints of zero mean and unit variance (to avoid the trivial constant solution). The temporal variation of the output y is defined as the average of its squared temporal

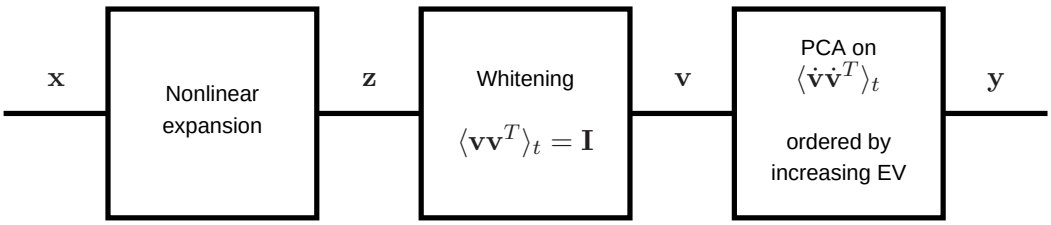

Figure 3.1: **Blockdiagram of SFA.** The algorithm is applied to a multi-dimensional time series $\mathbf{x}$. It consists of an optional expansion step which computes a number of fixed static nonlinear combinations $\mathbf{z}$ of the components of $\mathbf{x}$. Later in this Chapter we will show that this step can be performed by a cortical microcircuit of neurons. If this step is left out, $\mathbf{z} = \mathbf{x}$. In the next step, the expanded input $\mathbf{z}$ has to be whitened such that the components of the signal $\mathbf{v}$ have zero mean, unit variance, and are decorrelated. The final step selects from this whitened signal the direction of minimal temporal variation, i.e., the least principal component of the temporal derivative $\dot{\mathbf{v}}$. The projection onto this direction yields the slowest feature y_1. Multiple slow features $\mathbf{y} = (y_1, y_2, \ldots)$ are obtained from orthogonal projection directions which form the eigenvectors of the covariance matrix $\langle \dot{\mathbf{v}}\dot{\mathbf{v}}^T \rangle_t$, ordered by increasing eigenvalue.

derivative $\langle \dot{y}^2 \rangle_t$, where $\langle \cdot \rangle_t$ denotes averaging over time. In other words, SFA finds that function[1] g out of a certain predefined function space that produces the slowest possible output $y = g(\mathbf{x})$. This optimization problem is hard to solve in the general case (see Wiskott, 2003), but if the available function space is constrained to linear combinations of a whitened input, the problem has an elegant solution in the form of an eigenvalue problem in the covariance matrix of input time derivatives. More precisely, the slowest output is produced by the eigenvector of this matrix that corresponds to the *smallest* eigenvalue. This results in the standard SFA algorithm as presented in (Wiskott, 1998; Wiskott and Sejnowski, 2002) (see blockdiagram in Figure 3.1), which contains an optional expansion step that computes a number of fixed nonlinear combinations of the components of $\mathbf{x}$. Such nonlinear expansion boosts the power of any subsequent linear processing (like a kernel for support vector machines (Schölkopf and Smola, 2002)). This nonlinear expansion enables SFA to effectively choose from a much larger set of functions g (containing also nonlinear projections from $\mathbf{x}$), even if the last processing step in the blockdiagram of Figure 3.1 is constrained to be linear.

The restriction to linear functions in the last step of SFA allows that this processing step could in principle be carried out in a biological neural system by readout or projection neurons that extract information from a cortical microcircuit. A linear function is a reasonable approximation to the expressive capability of a readout neuron. The last step of SFA, the selection of the least principal component of the input time derivatives, could in principle be solved by anti-Hebbian learning on the differential input and output signals (Mitchison, 1991). Furthermore (Sprekeler et al., 2007) have shown that this is equivalent to choosing the principal component of a low-pass filtered input, which can in principle be solved by standard Hebbian

[1]Note that this function is a static input-output mapping $y(t) = g(\mathbf{x}(t))$, which at any time t transforms the input $\mathbf{x}(t)$ into an output value $y(t)$ instantaneously.

learning. In addition they have shown that an experimentally supported synaptic plasticity rule, spike-timing-dependent plasticity (STDP), could in principle enable spiking neurons to learn this processing step without supervision, provided that the presynaptic inputs are preprocessed in a suitable manner. This result suggests, that the gap between the abstract SFA learning principle for linear neurons that we examine in this Chapter and neurophysiological data on synaptic plasticity could eventually be closed. However, this analysis leaves open the question how the first two processing stages could be carried out by a biological neural system. We will show that a standard model for a generic cortical microcircuit could carry out the first processing step in the diagram of Figure 3.1 for the case where the time series $\mathbf{x}$ consists of multiple low-pass filtered spike trains. The question remains how the second processing step of Figure 3.1, the whitening, could be implemented by a neural circuit. Several learning methods that achieve whitening through a network of neurons have been proposed (Goodall, 1960; Atick and Redlich, 1993) (see chapter 8 of Dayan and Abbott, 2001). There also exist experimental data which suggest that the response of cortical neurons to natural external stimuli tends to be quite decorrelated (see e.g., Vinje and Gallant, 2000).

We establish in section 3.2 a relationship between the unsupervised SFA learning method and a commonly used learning method for supervised classification learning: the Fisher linear discriminant (FLD). More precisely, we show that SFA approximates the discrimination capability of the FLD in the sense that both methods yield the same projection direction, which can be interpreted as a separating hyperplane in the input space. This approximation holds for a simple condition on the temporal statistics of the input time series to SFA: The probability that two successive samples are from different classes has to be low. Through its tendency to produce a slowly varying output, SFA automatically clusters those inputs together that often occur in immediate consecution, and classifies them as different samples from the same category.

SFA is a learning method that does not require explicit supervision in the sense that the input patterns are given together with the target classification (labels). We show instead that it suffices to provide SFA with a very weak or implicit supervisor in the sense that successive input patterns tend to belong to the same class. (Li and DiCarlo, 2008) have referred to this as "unsupervised temporal slowness learning" and for brevity we use the term *unsupervised learning* in this Chapter.

SFA may also elucidate a puzzle regarding internal codes and computational mechanisms of the brain. A number of experimental data have challenged the classical view of coding and computation in the brain, which was based on the assumption that external stimuli and internal memory items are encoded by firing states of neurons, which assign a certain firing rate to a number of neurons that is maintained for some time interval. This classical view of neural coding has the advantage that one can apply a variety of readily available computational models from computer science and artificial neural networks in order to model computation in the brain. However, numerous recent experimental data suggest that many types of natural sensory stimuli, as well as internally generated traces for episodic memory, are encoded by characteristic trajectories (or sequences) of different firing states of neurons that stretch over several hundred ms. This result has been found

both for (seemingly) static external stimuli such as odors (Mazor and Laurent, 2005; Broome et al., 2006) (see Rabinovich et al., 2008, for a review) and tastes (Jones et al., 2007), and for intrinsically time-varying external stimuli such as natural auditory and visual stimuli (see Buonomano and Maass, 2009, for a review). In addition, numerous experimental data on replay of episodic memory from hippocampus to cortex point to sequences of different firing states, rather than single firing states of networks of neurons, as a common form of traces of episodic memory in hippocampus and cortex (see e.g., Euston et al., 2007; Ji and Wilson, 2008). These experimental data give rise to the question, how the brain can compute with such temporally dispersed information in the form of trajectories of firing states. At the latest at the top-level of information processing in the brain, where percepts are formed and decisions are made, the ubiquitous distribution of salient information over a sequence of different firing states (stretching over several hundred ms) has to be inverted, and compressed into a much shorter time interval. The theoretical analysis provided in this Chapter explains why, and under what conditions, this is possible with SFA learning.

In section 3.3 we test the theoretically predicted emergent discrimination capability of SFA by applying it to the output of a simulated network of spiking neurons, more precisely, a detailed model for a laminar cortical microcircuit (Häusler and Maass, 2007) based on data from (Thomson et al., 2002) and from the lab of Markram (Gupta et al., 2000). We injected spike trains that simulate the response of the cochlea to different spoken digits as inputs to the simulated cortical microcircuit, and examined whether linear readouts that receive as input a whitened version of the continuously varying firing response (in the form of low-pass filtered spike trains) from neurons in this circuit can learn without supervision to discriminate between different spoken digits. This experiment turned out to be successful, and it also revealed a possible functional advantage of this processing scheme: Linear readout neurons learned not only without supervision to discriminate between different spoken digits, but they provided correct predictions of the currently spoken digit already while the digit was still being spoken. This is what we refer to as "anytime computing": An "anytime computation" is a special form of an online computation, which can be prompted at any time to provide its current best guess of a proper output, by integrating as much information about previously arrived input pieces as possible. In another experiment, SFA was able to both detect and identify spike patterns within a continuous stream of Poisson input. Again this information was available already during the presentation of a pattern. This feature, which is predicted by the theoretical analysis of SFA learning, could enable subsequent processing stages in the brain to begin higher level computational processing already before the trajectory of network states that is characteristic for a particular sensory stimulus has ended. This feature might remove one obstacle for establishing a computational model for hierarchical processing of sensory information in the cortex: if each stage waits with its processing until the trajectory of firing states in the lower area has ended, and then creates a subsequent trajectory as a result of its own computational processing, the resulting total computation time becomes too long. If however readout neurons can transmit "at any time" their current guess regarding the identity of the circuit input, other areas to which

these readout neurons project can start their computational processing right away. During the subsequent few hundreds of ms they could in addition collect further evidence which they will receive from the same readout neurons, for or against the initial guess. In this computational paradigm the stream of sensory stimuli could generally be processed in real time, with significant processing delays arising only in the case of ambiguous sensory stimuli.

3.2 A theoretical basis for the emergent discrimination capability of SFA

In this section, we first give a definition of SFA and FLD. We then present a criterion on the temporal statistics of training examples which clarifies when SFA approximates FLD. Finally, we show how the SFA objective is influenced by applying it to a sequence of whole trajectories of points instead of just to a sequence of individual training examples.

Slow feature analysis (SFA) SFA extracts the slowest component y from a multi-dimensional input time series $\mathbf{x}$ by minimizing the temporal variation $\Delta(y)$ of the output signal y (Wiskott and Sejnowski, 2002),

$$\min \quad \Delta(y) := \langle \dot{y}^2 \rangle_t, \tag{3.1}$$

under the additional constraints of zero mean ($\langle y \rangle_t = 0$) and unit variance ($\langle y^2 \rangle_t = 1$). The notation $\langle \cdot \rangle_t$ is used in this Chapter to denote averaging over time. If multiple slow features are extracted an additional constraint ensures that they are decorrelated ($\langle y_i y_j \rangle_t = 0$) and ordered by decreasing slowness.

If we assume that the time series $\mathbf{x}$ has zero mean ($\langle \mathbf{x} \rangle_t = \mathbf{0}$) and if we only allow linear functions $y = \mathbf{w}^T \mathbf{x}$ the problem simplifies to the following objective

$$\min \quad J_{SFA}(\mathbf{w}) := \frac{\mathbf{w}^T \langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x} \mathbf{x}^T \rangle_t \mathbf{w}}. \tag{3.2}$$

The matrix $\langle \mathbf{x} \mathbf{x}^T \rangle_t$ is the covariance matrix of the input time series and $\langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t$ denotes the covariance matrix of time derivatives of the input time series (or time differences, for discrete time). The weight vector $\mathbf{w}$ which minimizes the quotient in (3.2) is the solution to the generalized eigenvalue problem

$$\langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \mathbf{w} = \lambda \langle \mathbf{x} \mathbf{x}^T \rangle_t \mathbf{w} \tag{3.3}$$

corresponding to the smallest eigenvalue λ. That is, we consider only the linear part of SFA here and ignore the nonlinear expansion step in Figure 3.1 for the moment (i.e., $\mathbf{z} = \mathbf{x}$). Note that the whitening step is made implicit here in the formulation of (3.2), like in (Berkes and Wiskott, 2003).

Fisher's linear discriminant (FLD) The FLD is a different data analysis method. It is applied to single data points $\mathbf{x}$, rather than time series. Furthermore it requires *labeled* training examples $\langle \mathbf{x}, c \rangle$, where $c \in \{1, \cdots, C\}$ is the class

to which this example belongs (we will first focus on the case $C = 2$). Hence it is a method for *supervised* learning. The goal is to find a weight vector $\mathbf{w}$ so that the class of new (unlabeled) test examples can be predicted from the value of $\mathbf{w}^T\mathbf{x}$ (predicting that $\mathbf{x}$ belongs to class 2 if $\mathbf{w}^T\mathbf{x} \geq \theta$ for some threshold θ, else that $\mathbf{x}$ belongs to class 1).

FLD searches for that projection direction $\mathbf{w}$ which maximizes the separation between classes while at the same time minimizing the variance within classes, thereby minimizing the class overlap of the projected values:

$$\max \quad J_{FLD}(\mathbf{w}) := \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}. \tag{3.4}$$

For two point sets S_1 and S_2 with means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, $\mathbf{S}_B$ is the between-class covariance matrix given by the separation of the class means

$$\mathbf{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T, \tag{3.5}$$

and $\mathbf{S}_W$ is the within-class covariance matrix given by

$$\mathbf{S}_W = \sum_{\mathbf{x}\in S_1} (\mathbf{x} - \boldsymbol{\mu}_1)(\mathbf{x} - \boldsymbol{\mu}_1)^T + \sum_{\mathbf{x}\in S_2} (\mathbf{x} - \boldsymbol{\mu}_2)(\mathbf{x} - \boldsymbol{\mu}_2)^T. \tag{3.6}$$

Again, the vector $\mathbf{w}$ optimizing (3.4) can be viewed as the solution to a generalized eigenvalue problem,

$$\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w}, \tag{3.7}$$

corresponding to the *largest* eigenvalue λ. Figure 3.3A illustrates the idea of FLD. It finds that direction $\mathbf{w}$ that optimizes the separability between the projected values of different classes S_1 and S_2 by additionally taking into account the within-class variances. Choosing the direction $\mathbf{w}'$ that only maximally separates the class means results in an overlap of the projected values. The FLD had been introduced in (Fisher, 1936). Good descriptions can be found in (Duda et al., 2000; Bishop, 2006).

3.2.1 Application to a classification problem with two classes

SFA and FLD receive different data types as inputs: unlabeled time series for SFA, in contrast to labeled single data points $\langle \mathbf{x}, c \rangle$ for the FLD during training, and unlabeled single data points $\mathbf{x}$ during evaluation of its resulting generalization capability after training.

Therefore, in order to apply the unsupervised SFA learning algorithm to the same classification problem as the supervised FLD, we have to convert the labeled training samples into a time series of unlabeled data points that can serve as an input to the SFA algorithm. In the following we create such a training time series from the classification problem by choosing at each time step a particular point from $S_1 \cup S_2$. We investigate the relationship between the weight vector found by Fisher's linear discriminant on the original classification problem and the weight vector found by slow feature analysis applied to the resulting training time series. The idea is that if we create the time series in such a way that most of its transitions, i.e., pairs of

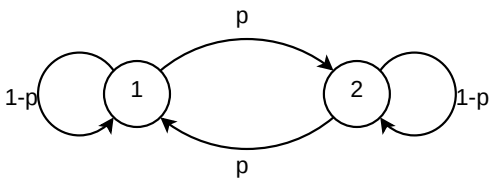

Figure 3.2: **Markov model describing the generation of the input time series to SFA from a two-class FLD problem.** The state c corresponds to the class S_c from which the current point in the time series is drawn. After the selection of each point the class of the next point is determined according to the transition probabilities between the states.

consecutive points, consist of point pairs from the same class, SFA should learn to be invariant to points from the same class and to extract the hidden class label of data points as a slowly varying feature of the time series.

First, we consider a classification problem with 2 classes, i.e., assume we are given two point sets $S_1, S_2 \subset \mathbb{R}^n$,

$$S_1 := \{\mathbf{x}_i^1 | i = 1, \ldots, N\}, \tag{3.8}$$

$$S_2 := \{\mathbf{x}_j^2 | j = 1, \ldots, N\}, \tag{3.9}$$

where $\mathbf{x}_i^1$ and $\mathbf{x}_j^2$ denote the data points of class 1 and 2, respectively (note that these points are unlabeled, since the superscripts 1 and 2 are not "visible" for the algorithms; it may also occur that $\mathbf{x}_i^1 = \mathbf{x}_j^2$). For simplicity we assume that both sets are of the same size N. We choose the following Markov model (see Figure 3.2) to create a time series $\mathbf{x}_t$ out of these two point sets S_1 and S_2: First, we choose one of the two classes with equal probability. Then we select a random point from the corresponding set (S_1 or S_2). This is then the first point in the input time series, $\mathbf{x}_1$. Next, we switch the class with a certain probability p (or leave it unchanged with probability $1-p$) and choose a point from the resulting class as the next input point, $\mathbf{x}_2$. This is repeated until the time series has a certain predefined length T. The states in the underlying Markov model correspond to the class from which the data point is currently drawn. After each drawing, the class is either switched with probability p, or left unchanged with probability $1-p$. The stationary distribution of this Markov model is

$$\boldsymbol{\pi} = \left(\frac{1}{2}, \frac{1}{2}\right). \tag{3.10}$$

Because we have chosen the initial distribution $\mathbf{p}_0 = \boldsymbol{\pi}$ we can say that at any time the current point is drawn from class 1 or class 2 with probability 1/2.

In this case we can express the matrices $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ of the SFA objective (3.2) in terms of the within-class and between-class scatter matrices of the FLD (3.4), $\mathbf{S}_W$ and $\mathbf{S}_B$ (for a derivation see Appendix C.1.1):

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{2N}\mathbf{S}_W + \frac{1}{4}\mathbf{S}_B, \tag{3.11}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{1}{N}\mathbf{S}_W + p \cdot \mathbf{S}_B. \tag{3.12}$$

Note that only $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ depends on p, whereas $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ does not.

For small p we can neglect the effect of $\mathbf{S}_B$ on $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ in (3.12). In this case the time series consists mainly of transitions within a class, whereas switching between the two classes is relatively rare. Therefore the covariance of time derivatives is mostly determined by the within-class scatter of the two point sets, and both matrices become approximately proportional: $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \approx 1/N \cdot \mathbf{S}_W$. Moreover, if we assume that $\mathbf{S}_W$ (and therefore $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$) has only nonzero eigenvalues, we can rewrite the SFA objective as

$$\begin{aligned} \min \quad J_{SFA}(\mathbf{w}) &\Leftrightarrow \max \quad \frac{1}{J_{SFA}(\mathbf{w})} \\ &\Leftrightarrow \max \quad \frac{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}}{\mathbf{w}^T \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{w}} \\ &\overset{(3.11),(3.12)}{\Leftrightarrow} \max \quad \frac{1}{2} + \frac{N}{4} \cdot \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \\ &\Leftrightarrow \max \quad J_{FLD}(\mathbf{w}). \end{aligned} \tag{3.13}$$

In the third line we inserted the expressions for $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ (3.11) and the approximation for $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ (3.12) for small p. That is, in this case where switching between different classes is rare compared to transitions within a class, the weight vector that yields the slowest output function is approximately equal to the weight vector that is optimal in separating the two classes in the sense of FLD.

Figure 3.3 demonstrates this relationship on a sample two-class problem in two dimensions. We interpret the weight vectors found by both methods as normal vectors of hyperplanes in the input space. Since an additional bias value is required to determine a unique hyperplane for each weight vector, we place the hyperplanes in Figure 3.3B simply onto the mean value[2] $\boldsymbol{\mu}$ of all training data points (i.e., the hyperplanes are defined as $\mathbf{w}^T\mathbf{x} = \theta$ with $\theta = \mathbf{w}^T\boldsymbol{\mu}$). One sees that the weight vector found by the application of SFA to the training time series $\mathbf{x}_t$ generated with $p = 0.2$ is approximately equal to the weight vector resulting from FLD on the initial sets of training points. The deviation comes from the fact that the covariance matrix of time differences, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$, is not solely determined by the within-class scatter (in eq. (3.12)), because the time series switches several times between the classes.

We interpret the slowest feature found by the SFA algorithm as the hypothesis of a linear classifier ($h(\mathbf{x}) = \text{sign}(\mathbf{w}_{SFA}^T(\mathbf{x} - \boldsymbol{\mu}))$). Figure 3.3C shows the prediction of this hypothesis for unseen test points from each class, drawn from the same distribution as the training point sets S_1 and S_2. It can be seen that the output of the slowest feature of this test time series (which corresponds just to the projection of its points onto the weight vector $\mathbf{w}_{SFA}$) takes on distinct values for different classes. This demonstrates that SFA has extracted the class of the points as the slowest varying feature by finding a direction that separates both classes, and that this ability generalizes to test points not used for training.

[2]Note that for this particular choice of time series generation the expected mean of the training time series is equal to the total mean of the training data points. Since SFA subtracts the mean of the training time series beforehand, this value is mapped to 0 in the SFA output.

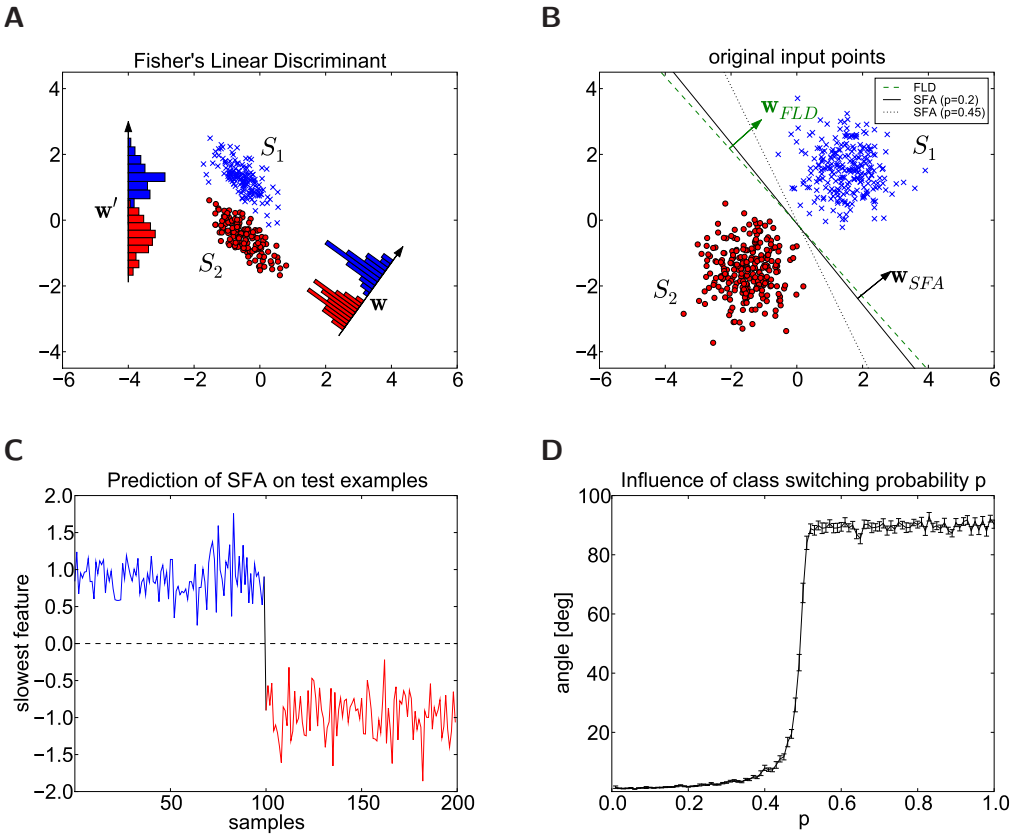

Figure 3.3: (see next page for Figure caption)

Figure 3.3D quantifies the deviation of the weight vector resulting from the application of SFA to the time series from the one found by FLD on the original points. We use the angle between both weight vectors as an error measure. For each value of p we generate 100 random classification problems such as the one shown in Figure 3.3B and calculate the average angle between the vectors obtained by both methods on these problems (see Appendix C.2.1 for details). Since the sign of the vectors is arbitrary, we always took the smaller of the two possible angles. Thus, an angle of 0° means perfect equivalence, and the maximal achievable angle (i.e., error) is 90°. It can be seen that if p is low, i.e., transitions between classes are rare compared to transitions within a class, the angle between the vectors is small and SFA approximates FLD very well. The angle increases moderately with increasing p; even with higher values of p (up to 0.45) the approximation is reasonable and a good classification by the slowest feature can be achieved. As soon as p reaches a value of about 0.5, the error grows almost immediately to the maximal value of 90°. It can be seen from equations (3.11) and (3.12) that for $p = 0.5$ the covariance of time derivatives, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$, becomes proportional to the covariance of the input, $\langle \mathbf{x}\mathbf{x}^T \rangle_t$, which means that every possible vector $\mathbf{w}$ is a solution to the generalized eigenvalue problem (3.3), resulting in an average angle of about 45°. With $p = 0.5$

Figure 3.3: **Relationship between unsupervised SFA and supervised FLD for a two-class problem in 2D.** (**A**) Illustration of the concept of FLD. Shown are two point sets and histograms of values obtained by projecting points onto two different directions, $\mathbf{w}$, the direction resulting from FLD, and $\mathbf{w}'$, the direction of maximal separation of the class means. (**B**) Sample point sets with 250 points for each class, drawn from two different Gaussian distributions. The green arrow and the green dashed line indicates the weight vector ($\mathbf{w}_{FLD}$) and a corresponding hyperplane, respectively, resulting from the application of FLD to the two-class problem. The black arrow and the black solid line shows the weight vector ($\mathbf{w}_{SFA}$) and a hyperplane resulting from SFA applied to the time series generated from these training points as described in the text ($T = 5000$, $p = 0.2$). The black dotted line displays an additional SFA hyperplane resulting from a time series generated with $p = 0.45$. All hyperplanes are placed onto the mean value of all training points. (**C**) Output of the SFA algorithm (slowest feature) applied to a test time series consisting of 100 points from class 1 (blue) and 100 points from class 2 (red; colors as in **B**). These test points were drawn from the same Gaussian distributions as in **B**, but were not used for training. Each value of the trace corresponds to a projection of a point onto the weight vector of the slowest feature ($\mathbf{w}_{SFA}$ in **B**). The dashed line at 0 corresponds to points on the solid SFA-hyperplane shown in **B**. (**D**) Dependence of the error between the weight vectors found by FLD and SFA on the switching probability p. This error is defined as the average angle between the weight vectors obtained on 100 randomly chosen classification problems. Error bars denote the standard error of the mean. Good approximations can still be achieved with rather high values of p (up to 0.5).

points are drawn randomly from the union of the two point sets, independently of the class previously chosen, i.e., the class information is neglected altogether. For values of $p > 0.5$ switching between classes becomes so frequent that SFA cannot extract the class information anymore resulting in vectors orthogonal to the FLD vector.

3.2.2 Application to classification problems with more than two classes

The results in the previous section can also be extended to the case of C classes ($C > 2$), showing the equivalence between the space spanned by the $C - 1$ slow features extracted by SFA and the $C - 1$-dimensional subspace resulting from the application of a generalized version of Fisher's linear discriminant (Duda et al., 2000).

Again, we start from a classification problem with C disjoint point sets $S_c \subset \mathbb{R}^n, c = 1, \ldots, C$,

$$S_c := \{\mathbf{x}_i^c | i = 1, \ldots, N_c\}, \tag{3.14}$$

where $\mathbf{x}_i^c$ denote the data points of class c. In contrast to the previous section we consider here the more general case that the number of points in each class is different. Let N_c denotes the number of data points in class c, and let $N_T = \sum_{c=1}^{C} N_c$ be the total number of points. From these point sets we generate a time series $\mathbf{x}_t$ analogously as in the previous section, using a generalization of the Markov model in Figure 3.2 with C states $S = \{1, 2, \ldots, C\}$. We define the transition probability

from state $i \in S$ to state $j \in S$ as

$$P_{ij} = \begin{cases} a \cdot \frac{N_j}{N_T} & \text{if } i \neq j, \\ 1 - \sum_{k \neq j} P_{ik} & \text{if } i = j, \end{cases} \tag{3.15}$$

with some appropriate constant[3] $a > 0$. It is easy to show (see Appendix C.1.2) that

$$\boldsymbol{\pi} = \left(\frac{N_1}{N_T}, \frac{N_2}{N_T}, \ldots, \frac{N_C}{N_T} \right) \tag{3.16}$$

is a stationary distribution of this Markov model. This means that the probability that any point in the time series is chosen from a particular class is proportional to the size of the corresponding point set compared to the number of total points.

For this particular way of generating a time series from the input points we can calculate the following expressions for the covariance matrices of the input and time derivatives in terms of the within-class and between-class covariances (see Appendix C.1.2):

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{N_T}\mathbf{S}_W + \frac{1}{N_T}\mathbf{S}_B, \tag{3.17}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{2}{N_T}\mathbf{S}_W + \frac{2a}{N_T}\mathbf{S}_B. \tag{3.18}$$

Note that equations (3.17) and (3.18) are similar to (3.11) and (3.12). Again, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ depends on a, whereas $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ does not. Note that the commonly used definition for the between-class scatter matrix $\mathbf{S}_B$ (see e.g., Duda et al., 2000) for the multi-class case is slightly different from the two class case (3.5). For small a, i.e., when transitions between classes are rare compared to transitions within a class, we can approximate $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \approx 2/N_T \cdot \mathbf{S}_W$.

We recall the definition of SFA as a generalized eigenvalue problem (3.3) and insert (3.17) and (3.18) for negligible a:

$$\begin{aligned} \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{W} &= \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{W}\boldsymbol{\Lambda} \\ \overset{(3.17),(3.18)}{\Leftrightarrow} \quad \frac{2}{N_T}\mathbf{S}_W\mathbf{W} &= \frac{1}{N_T}\mathbf{S}_W\mathbf{W}\boldsymbol{\Lambda} + \frac{1}{N_T}\mathbf{S}_B\mathbf{W}\boldsymbol{\Lambda} \\ \Leftrightarrow 2\mathbf{S}_W\mathbf{W}\boldsymbol{\Lambda}^{-1} &= \mathbf{S}_W\mathbf{W} + \mathbf{S}_B\mathbf{W} \\ \Leftrightarrow \mathbf{S}_B\mathbf{W} &= \mathbf{S}_W\mathbf{W}\left[2\boldsymbol{\Lambda}^{-1} - \mathbf{E}\right], \end{aligned} \tag{3.19}$$

where $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_n)$ is the matrix of generalized eigenvectors and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ is the diagonal matrix of generalized eigenvalues. We used the assumption that $\mathbf{S}_W$ (and therefore $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$) are positive definite, i.e., all eigenvalues λ_i are strictly positive ($\boldsymbol{\Lambda}^{-1}$ exists). The last line of (3.19) is just the formulation of FLD as a generalized eigenvalue problem (see (3.7)). More precisely, the eigenvectors of the SFA problem are also eigenvectors of the FLD problem, i.e., the $C-1$ slowest features extracted by SFA applied to the time series $\mathbf{x}_t$ span the

[3] For $C = 2$ and $N_1 = N_2 = N_T/2$ the class is switched at each time with probability $p = a/2$ and left unchanged with probability $1 - p$.

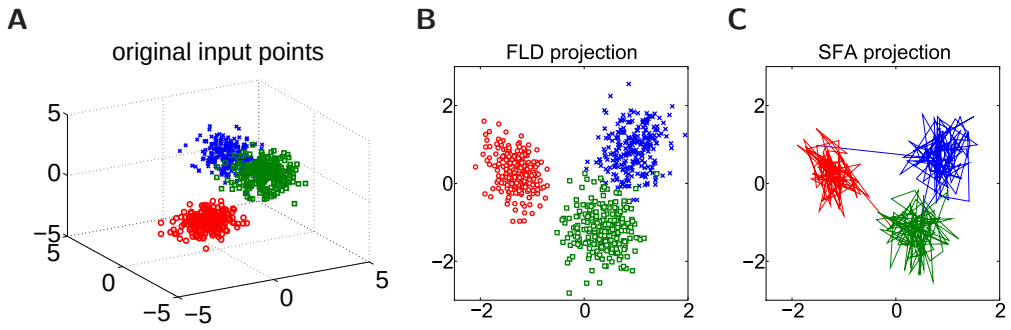

Figure 3.4: **Relationship between SFA and FLD for a three-class problem in 3D.** (**A**) Sample point sets with 250 points for each class, drawn from three different Gaussian distributions. (**B**) Point sets projected onto the 2-dimensional subspace found by FLD (colors and markers as in **A**). The FLD maximizes the between-class scatter while minimizing the within-class scatter. (**C**) Phase plot of the two slowest features found by SFA applied to a test time series consisting of 100 points from each class, which were drawn from the same Gaussian distributions as in **A**, but not used for training. The training sequence for SFA was generated from the input points in **A** as described in the text ($T = 5000$, $a = 0.5$). This corresponds to a projection of these test points onto the subspace spanned by the two slowest features. The color encodes the class of the respective point in the test sequence (colors as in **A**, **B**). Note the similarity between panels **B** and **C**.

subspace that optimizes separability in terms of Fisher's linear discriminant. Note that the eigenvalues correspond by

$$\lambda_i^{FLD} = \frac{2}{\lambda_i^{SFA}} - 1, \tag{3.20}$$

which means the order of eigenvalues is reversed, since all eigenvalues λ_i^{SFA} are positive. The slowest feature (corresponding to the smallest eigenvalue in the first line of (3.19)) is the weight vector which achieves maximal separation (largest eigenvalue in the last line of (3.19)).

This similarity of the subspace found by FLD on the initial point sets and by SFA on the time series is demonstrated in Figure 3.4. Panel B shows the projection of the data points shown in panel A onto the 2-dimensional subspace resulting from FLD, while Panel C plots the trajectory of the two slowest features found by SFA applied to a test time series generated from points drawn from the same distributions as the original points in panel A. One sees that both projections are almost identical, which means that the subspace that maximizes separability in terms of Fisher is equal to the subspace spanned by the slowest features of our particular time series. Note that there is more than one particular pair of directions which span the same 2-dimensional subspace. Therefore, while both methods extract the same subspace, the exact projections might look different (e.g., the signs of individual eigenvectors may be flipped, or the projections could be rotated against each other, if the eigenvalues are close to degenerate).

3.2.3 Application to trajectories of training examples

In sections 3.2.1 and 3.2.2 we have shown that SFA approximates the classification capability of FLD if the probability is low that two successive points in the input time series to SFA are from different classes. In order to generate a time series from the classification problems we chose at each time step the class of the points with a certain probability according to a Markov model, but apart from that class information, however, each point was chosen independently from the preceding point in the time series. The optimal response to such a time series is to produce a constant response during periods where only points from a single class are presented (see also Berkes, 2006). This approximately piecewise constant function will become more smooth as the size of the function space increases, but it will remain a step function. This classification capability of SFA relies on the fact that SFA sees each possible transition between two points from the same class approximately equally often, and therefore produces a similar output for each point from that class.

What happens if these time series consist of whole *trajectories* of single points, e.g., repeated occurrences of characteristic sequences of firing states in neural circuits? In this section we investigate how the SFA objective changes when the input time series consists of trajectories of points instead of individual points only.

3.2.3.1 Repetitions of a fixed trajectory

First, we consider a time series $\mathbf{x}_t$ consisting of multiple repetitions of a fixed predefined trajectory $\tilde{\mathbf{t}} := (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_{\tilde{T}})$ of $\tilde{T}$ n-dimensional points $\tilde{\mathbf{x}}_k$, which are embedded into noise input. Initially the trajectory points $\tilde{\mathbf{x}}_k$ are drawn from a certain distribution. Between any two repetitions of this trajectory noise input is presented, which consists of a random number of points drawn from the same distribution, but independently at each time step.

It is easy to show (see Appendix C.1.3) that for such a time series the SFA objective (3.2) reduces to

$$\min \quad J_{SFA}(\mathbf{w}) \Leftrightarrow \max \quad \frac{\mathbf{w}^T \tilde{\mathbf{\Sigma}}_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}}, \tag{3.21}$$

where

$$\tilde{\mathbf{\Sigma}}_t := \frac{1}{\tilde{T}-1} \sum_{k=2}^{\tilde{T}} \left(\tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_{k-1}^T + \tilde{\mathbf{x}}_{k-1} \tilde{\mathbf{x}}_k^T \right) \tag{3.22}$$

is the covariance matrix of the trajectory $\tilde{\mathbf{t}}$ with $\tilde{\mathbf{t}}$ delayed by one time step, i.e., it measures the temporal covariances (hence the index t) of $\tilde{\mathbf{t}}$ with time lag 1. Such time-delayed correlation matrices have also been introduced in (Blaschke et al., 2006, 2007) to show the relationship between SFA and second-order ICA. Note that in the standard classification problems described previously the time series $\mathbf{x}_t$ had no temporal correlations apart from the class information at all, i.e., consecutive points were uncorrelated given their class labels.

That is, choosing the weight vector $\mathbf{w}$ that produces the slowest output is equivalent to choosing the vector that maximizes the temporal correlations of the output

during instances of the trajectory $\tilde{\mathbf{t}}$. In other words, $\mathbf{w}$ is the (generalized) eigenvector of $\tilde{\mathbf{\Sigma}}_t$ which corresponds to the largest eigenvalue of this matrix. Since the transitions between two successive points of the trajectory $\tilde{\mathbf{t}}$ occur much more often in the time series $\mathbf{x}_t$ than transitions between any other possible pair of points, SFA has to respond as smoothly as possible during $\tilde{\mathbf{t}}$ in order to produce the slowest possible output, whereas the average response to noise samples should ideally be zero. This means that SFA is able to detect these repetitions of $\tilde{\mathbf{t}}$ by responding during such instances with a distinctive shape.

Figure 3.5A shows the response of SFA, which was trained on a sequence of 100 repetitions of a fixed trajectory $\tilde{\mathbf{t}}$, interleaved with random intervals of noise input from the same distribution. It can be seen that during each instance of $\tilde{\mathbf{t}}$ SFA responds with the same smooth curve. Due to the intermittent noise input, this curve has to be cyclic and have zero mean. Typically this response is similar to a section of a sine wave, which is theoretically the slowest possible response for the general SFA optimization problem (3.1) (Wiskott, 2003). The smoothness of this sine wave critically depends on the number of trajectory repetitions (the proportion of time trajectories are presented compared to noise), the dimensionality of the state space, and the complexity of the function space (which is constrained to be linear here). For display purposes we have chosen an overfitting regime in Figure 3.5A, since the dimensionality of the state space is larger than the length of the trajectory. In this example, SFA also responds with an increased amplitude during trajectory presentations. This can be explained by the fact that the slowest signal with a constrained variance is one which distributes this variance to times when it varies more slowly (i.e., during trajectory repetitions).

3.2.3.2 Several classes of trajectories

Next, we consider a classification problem given by two sets of trajectories, $\mathcal{T}_1, \mathcal{T}_2 \subset (\mathbb{R}^n)^{\tilde{T}}$, i.e., the elements of each set $\mathcal{T}_c$ are sequences of $\tilde{T}$ n-dimensional points[4]. We assume that all those points are distinct and that $\mathcal{T}_1$ and $\mathcal{T}_2$ are of the same size N. Moreover, we emphasize that we draw the trajectories from distributions with different means, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, as we did in the point discrimination examples. We generate a time series according to the same Markov model as in Figure 3.2. However, we do not choose individual points at each time step; rather we generate a sequence of trajectories: Initially, we choose a class from which the first trajectory is drawn, $\mathcal{T}_1$ or $\mathcal{T}_2$, and draw a random trajectory from this set. After each trajectory, we select a new trajectory of points after the previous one has ended. The class of this new trajectory is determined according to the transition probabilities in Figure 3.2.

For this time series consisting of such a trajectory sequence we can now express

[4]The generalization to C classes is analogous to section 3.2.2.

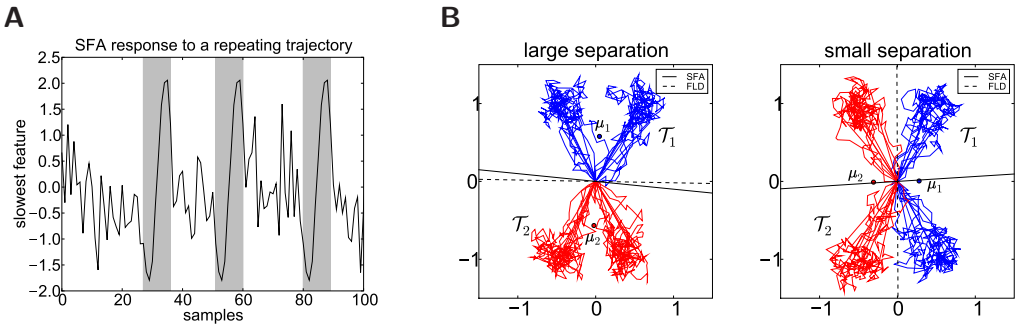

Figure 3.5: **Relationship between SFA and FLD for time series consisting of trajectories.** (**A**) SFA response to a time series consisting of a single repeating trajectory of training examples. A trajectory $\tilde{\mathbf{t}}$ is generated by randomly selecting $\tilde{T} = 10$ points from the uniform distribution of binary vectors, $\{0,1\}^n$ ($n = 50$). Repetitions of this trajectory $\tilde{\mathbf{t}}$ (shaded areas) are interleaved with a random number (drawn uniformly between 10 and 30) of individual single points drawn from the same distribution. The input time series that was used for training SFA consisted of 100 such repetitions of $\tilde{\mathbf{t}}$; a sample SFA response with 3 repetitions is shown. (**B**) Classification problem with two classes of artificial trajectories in 2D (blue, $\mathcal{T}_1$, and red, $\mathcal{T}_2$), each consisting of 20 trajectories of 100 points. The class means are denoted by $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$. In both panels the trajectories were drawn from the same distribution, but in the left panel the class labels were chosen in order to yield a large separation between the class means, whereas in the right panel this separation is small. The dashed line indicates a hyperplane corresponding to the weight vector obtained by application of FLD to the individual points of the trajectories. The solid line is the hyperplane found by SFA on a random sequence of 1000 trajectories. Both hyperplanes are placed onto the mean value of the trajectories. Note that the result of SFA is independent of p (here, $p = 0.5$).

the matrices $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ of the SFA objective (3.2) as (see Appendix C.1.3)

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{2N\tilde{T}} \mathbf{S}_W + \frac{1}{4} \mathbf{S}_B, \tag{3.23}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{1}{N\tilde{T}} \mathbf{S}_W + \frac{p}{\tilde{T}} \cdot \mathbf{S}_B - \frac{\tilde{T}-1}{\tilde{T}} \cdot \tilde{\boldsymbol{\Sigma}}_t. \tag{3.24}$$

The matrices $\mathbf{S}_W$ and $\mathbf{S}_B$ describe here the within-class and between-class scatter of the FLD objective (3.4) applied to point sets S_1 and S_2, which are composed of the individual points of the trajectories in $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively. Note that the covariance matrix $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ in (3.23) is equal to the case where the time series was composed of individual points instead of trajectories (see equation (3.11)). However, the temporal correlations induced by the use of trajectories has an effect on the covariance of temporal differences $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ in (3.24) compared to (3.12). First, it additionally depends again on the temporal covariance matrix $\tilde{\boldsymbol{\Sigma}}_t$, which is in this case the average temporal covariance with time lag 1 of all available trajectories in $\mathcal{T}_1$ and $\mathcal{T}_2$. Second, the switching probability p enters with a factor $1/\tilde{T}$, which becomes apparent when noting that whenever a trajectory is selected, $\tilde{T}$ points from the same class are presented in succession. Thus the effective switching probability

is $p/\tilde{T}$. Note that for $\tilde{T} = 1$ and $\tilde{\mathbf{\Sigma}}_t = \mathbf{0}$ equations (3.11) and (3.12) follow as a special case.

Equations (3.23) and (3.24) suggest that even for a small value of p the objective of SFA cannot be solely reduced to the FLD objective, but rather that there is a trade-off between the tendency to separate trajectories of different classes (as explained by the relation between $\mathbf{S}_B$ and $\mathbf{S}_W$) and the tendency to produce smooth responses during individual trajectories (determined by the temporal covariance matrix $\tilde{\mathbf{\Sigma}}_t$):

$$\min \quad J_{SFA}(\mathbf{w}) = \frac{\mathbf{w}^T \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}} \approx \frac{\mathbf{w}^T \left[\frac{1}{N\tilde{T}} \mathbf{S}_W\right] \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}} - \frac{\tilde{T}-1}{\tilde{T}} \cdot \frac{\mathbf{w}^T \tilde{\mathbf{\Sigma}}_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}}, \quad (3.25)$$

where the approximation is valid if $p/\tilde{T}$ is small[5]. That is, the SFA objective can be written as the difference between two terms. The weight vector $\mathbf{w}$ which minimizes the first term is equal to the weight vector found by the application of FLD to the classification problem of the individual trajectory points (note that $\mathbf{S}_B$ enters (3.25) through $\langle \mathbf{x}\mathbf{x}^T \rangle_t$, cf. eq. (3.13)). The weight vector which maximizes the second term is the one which produces the slowest possible response during individual trajectories. The factor $(\tilde{T}-1)/\tilde{T}$ is the proportion of transitions between successive points in the time series that belong to the same trajectory. If the separation between the trajectory classes is large compared to the temporal correlations (i.e., the first term in (3.25) dominates for the resulting $\mathbf{w}$) the slowest feature will be similar to the weight vector found by FLD on the corresponding classification problem. On the other hand, as the temporal correlations of the trajectories increase, i.e., the trajectories themselves become smoother, the slowest feature will tend to favor exploiting this temporal structure of the trajectories over the separation of different classes (in this case, eq. (3.25) is dominated by the second term for the resulting $\mathbf{w}$).

In the point discrimination example SFA derives its classification capability from seeing each possible transition between two points approximately equally often. This is not the case anymore when a sequence of trajectories is presented: now there are pairs of points from the same class which have too few transitions between them because most transitions are not between randomly chosen points, but within pre-defined trajectories. Furthermore, since the effective switching probability of the classes of two consecutive trajectories is reduced to $p/\tilde{T}$, the SFA objective (3.25) becomes essentially independent of the switching probability p, if the trajectories are sufficiently long. This means, that the SFA output does not depend on the temporal order of the trajectories any more, rather, the result is completely determined by the set of trajectories used for training. That is, by using a time series consisting of trajectories instead of individual points one loses the possibility to control the classification problem to be learned by changing the temporal statistics of the input. All possible class labellings of a given set of trajectories lead to the same direction learned by SFA. The class labelling which is in this case approximated by SFA according to (3.25) is the one which has the maximal separability in terms of the FLD, i.e., the one which corresponds to a scatter $\mathbf{S}_W$ which minimizes the first

[5]Note that the values of both numerators are in the same range because $\tilde{\mathbf{\Sigma}}_t$ is already a normalized covariance matrix (3.22) whereas $\mathbf{S}_W$ (3.6) needs to be normalized by a factor $1/N\tilde{T}$.

term in (3.25). This is demonstrated in Figure 3.5B, which shows two classification problems with artificial trajectories chosen from the same distribution of points, but with different assignment of class labels: one with a large and one with a small separation between the means of the trajectory classes. It can be seen that while the FLD always finds a separating hyperplane, SFA always approximates that classification problem with the larger separation. However, even if the slowest feature is not able to separate the classes, later SFA components, which find orthogonal directions to the previous ones, might be useful. For example, in the right panel of Figure 3.5B the second slowest feature would find a separating hyperplane.

In theory, the optimal response of SFA in this trajectory example would again be a piecewise constant function. However, if we introduce zero or noise input between two trajectories the optimal response would be half sine waves during presentations of individual trajectories, which are the typical SFA responses shown in (Wiskott and Sejnowski, 2002; Wiskott, 2003). If the means of the trajectory classes (e.g., $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ in Figure 3.5B) are equal, there would be no effect to discriminate classes in terms of Fisher's linear discriminant, because the first term in (3.25) vanishes. However, the theoretical analysis in (Wiskott, 2003) predicts that even in that case of equal class means SFA still provides a certain discrimination capability through the decorrelation constraint of multiple slow features: a feature that responds with half sine waves of different amplitudes for different patterns also varies slowly and can still be decorrelated to other responses. Thus, with an infinite function space SFA always produces a feature that responds with a different amplitude for each individual pattern. That is, in general SFA will try to distinguish all trajectories, but if the available function space is limited it might respond with the same amplitude to all trajectories which are similar, i.e., belong to the same class.

3.2.4 When does linear separation of trajectories of network states suffice?

Linear SFA can at best achieve a linear separation of trajectories of points. Although linear separation of complex trajectories of points is difficult in low dimensions, mathematical arguments imply that linear separation of such trajectories becomes much easier in higher dimensions. Consider artificial trajectories which are simply defined as a sequence of random points drawn uniformly from the d-dimensional hypercube $[0,1]^d$. Each point in this space corresponds to the vector of firing activities of the d presynaptic neurons of a readout at a particular time t. Each linear readout neuron defines a hyperplane in this state space by the particular setting of its weights. It assigns values 1 for points on one side of this hyperplane and values 0 to points on the other side of the hyperplane. Two trajectories are called linearly separable if they lie on different sides of some hyperplane. Figure 3.6A shows an example of such a pair of linearly separable trajectories in 3 dimensions. However, such a perfect separation of randomly drawn trajectories is very unlikely in this low-dimensional space.

Figure 3.6B shows that the situation changes drastically if one moves to higher-dimensional spaces. The black curve indicates the probability that any two ran-

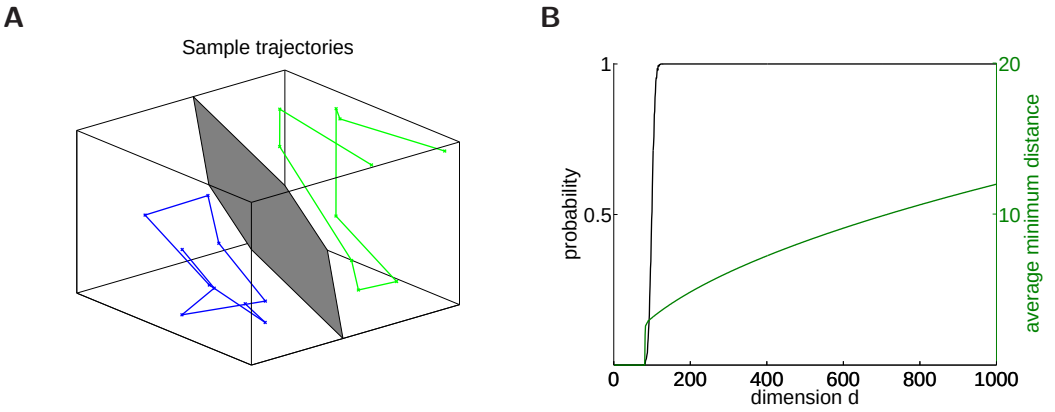

Figure 3.6: **Probability of linear separability increases with higher dimensionality of the state space.** (**A**) Two sample trajectories (green and blue curve) defined by connecting randomly drawn points from a 3-dimensional cube. These trajectories can be separated by a hyperplane (gray surface) defined by the synaptic weights of a linear discriminator. (**B**) Probability of linear separability of two randomly drawn trajectories of length 100 (black curve, left scale), and average minimum Euclidean distance between any two points of these two trajectories (green curve, right scale), as a function of the dimension d. Trajectories are defined as a sequence of random points drawn uniformly from the d-dimensional unit cube.

domly drawn trajectories of length 100 (i.e., each trajectory is defined by connecting 100 random points drawn uniformly from the d-dimensional unit cube) are linearly separable in d dimensions, for different values of d (see Appendix C.2.2). One sees that as soon as the dimension grows beyond 100, any two such trajectories become linearly separable with almost 100% probability. This holds for any length l of trajectories: for $d = l$, the probability of separation is 0.5 (see also Cover, 1965), if $d > l$ the probability converges very fast to 1. In other words, a linear readout neuron with d presynaptic inputs can separate almost any pair of trajectories that are each defined by connecting less than d randomly drawn points.

The green curve in Figure 3.6B shows the average of the minimal distance between such a pair of trajectories, which is defined as the minimal Euclidean distance between any point of trajectory 1 and any point of trajectory 2. This distance also grows with increasing dimension[6]. Thus, at higher dimensions d, it is not only more likely that any two trajectories of the length $l < d$ can be separated by a linear readout, but they can also be separated with an increasing "safety-margin" from the hyperplane. This implies that noisy variations of the same trajectories can be correctly classified by the same linear readout, which hints to a better generalization capability of linear readout neurons for higher dimensions.

[6]Note that the length of the main diagonal of a d-dimensional hypercube, i.e., the largest possible distance between any two points from the hypercube, is $\sqrt{d}$.

3.3 Application to unsupervised training of linear readouts from a cortical microcircuit model

In the previous section we have shown that slow feature analysis can directly be used for unsupervised linear discrimination of different point sets, if a time series is generated from these point sets in a way that the class is a slowly varying feature. Furthermore, we have shown how this property is affected if this time series consists of a sequence of trajectories instead of individual points. Now we turn our attention to SFA as a possible mechanism for training readouts of a biological microcircuit. The sequence of states that such a recurrent network undergoes in response to a specific stimulus forms a trajectory in state space. When presented with a sequence of such trajectories SFA should again be able to extract information about the stimulus in a similar way.

We have argued in the section 3.2.3 that the application of SFA to such a sequence of trajectories of network states differs from the application to individual points of a classification problem. In the latter case, a different input pattern has been presented at every single time step, whereas in the former case a single trajectory forms a sequence of input patterns from the same class. Due to the temporal correlations of these trajectories we do not expect that the slowest feature always perfectly extracts the class of the trajectories, as it did for the example with individual points in Figure 3.3. Rather, we predict that the class information will be distributed over multiple slow features. If multiple slow features are extracted, the feature y_i is the slowest feature under the additional constraint to be decorrelated to all slower features $y_1, \ldots, y_{i-1}$. This means that the slowest features are ordered by decreasing slowness, i.e., y_1 is the slowest feature, y_2 is the second slowest feature, and so on. In the following, features y_i with a higher index i are also called "higher order" features.

When computing with state trajectories in order to be able to extract reliable information about the stimulus, we want readouts of the circuit to produce an informative output not only at the end of the trajectory, but already while the trajectory is still being presented to the readout. Furthermore, this output should be as temporally stable as possible throughout the duration of a trajectory, hence providing an "anytime classification" of the stimulus. This requirement of temporal stability renders SFA a promising candidate for training readouts in an unsupervised fashion to discriminate "at any time" between trajectories in response to different stimulus classes. In the following we will discuss several computer simulations of a cortical microcircuit of spiking neurons where we trained a number of linear SFA readouts[7] on a sequence of network state trajectories, each of which is defined by the low-pass filtered spike trains of those neurons in the circuit that provide synaptic input to the readout neuron. Such recurrent circuits typically provide a temporal integration of the input stream and project it nonlinearly into a high-dimensional space (Maass et al., 2002), thereby boosting the expressive power of the subsequent linear SFA readouts. In the setup of Figure 3.1 the circuit therefore provides the map-

[7] We interpret the linear combination defined by each slow feature as the weight vector of a hypothetical linear readout.

ping from the inputs $\mathbf{x}$ to the expanded signals $\mathbf{z}$, i.e., the trajectories of network states. The readouts then compute the slowest features $\mathbf{y}$ from these trajectories. Note, however, that the whitening step is performed implicitly in the SFA optimization (3.2). As a model for a cortical microcircuit model we use the laminar circuit from (Häusler and Maass, 2007) consisting of 560 spiking neurons organized into layers 2/3, 4, and 5, with layer-specific connection probabilities obtained from experimental data (Gupta et al., 2000; Thomson et al., 2002).

3.3.1 Detecting embedded spike patterns

In the first experiment we investigated the ability of SFA to detect a repeating firing pattern within noise input of the same firing statistics. We recorded circuit trajectories in response to a sequence of 200 repetitions of a fixed spike pattern which are embedded into a continuous Poisson input stream. The input to the circuit consisted of 10 input spike trains. The pattern itself is defined as fixed Poisson spike trains of length 250ms and of rate 20Hz, the same rate as the background Poisson input (in the following also called noise input). We then trained linear SFA readouts on the 560-dimensional circuit trajectories, defined as the low-pass filtered spike trains of the spike response of all 560 neurons of the circuit (we used an exponential filter with $\tau = 30$ms and a sample time of 1ms). The period of Poisson input in between two such patterns was also randomly chosen; it was drawn uniformly between 100ms and 500ms.

Figure 3.7A shows a sample test stimulus consisting of a sequence of four pattern instances interleaved by random intervals of noise input, as well as the circuit response to this test stimulus and the 5 slowest features, y_1 to y_5, in response to the trajectory obtained by low-pass filtering this circuit response. At first glance, no clear difference can be seen between the raw SFA responses during periods of pattern presentations and during phases of noise input. The slow features are of course nonzero during noise input since the circuit response is quite similar to the response during patterns. However, we found that if we take the mean over the responses of multiple different noise phases, the average SFA output cancels away whereas a characteristic response remains during pattern presentations (see Figure 3.7C). This effect is predicted by the theoretical arguments in section 3.2.3 and can to some extent be seen in phase plots of traces that are obtained by a leaky integration of the slowest features in response to a test sequence of 50 embedded patterns (see Figure 3.7B). The slowest features span a subspace where the response during pattern presentations can be nicely separated from the response during noise input. Concerning this separability, SFA yields a significant improvement over randomly chosen linear functions, as shown in Figure 3.7D. That is, by simple threshold operations on the low-pass filtered versions of the slowest features one can in principle detect the presence of patterns within the continuous input stream. Furthermore, this extracted information is not only available after a pattern has been presented, but already during the presentation of the pattern, which supports the idea of "anytime computing".

One interesting property of this setup is that if we apply SFA directly on the stimulus trajectories, we basically achieve the same result. In fact, the application

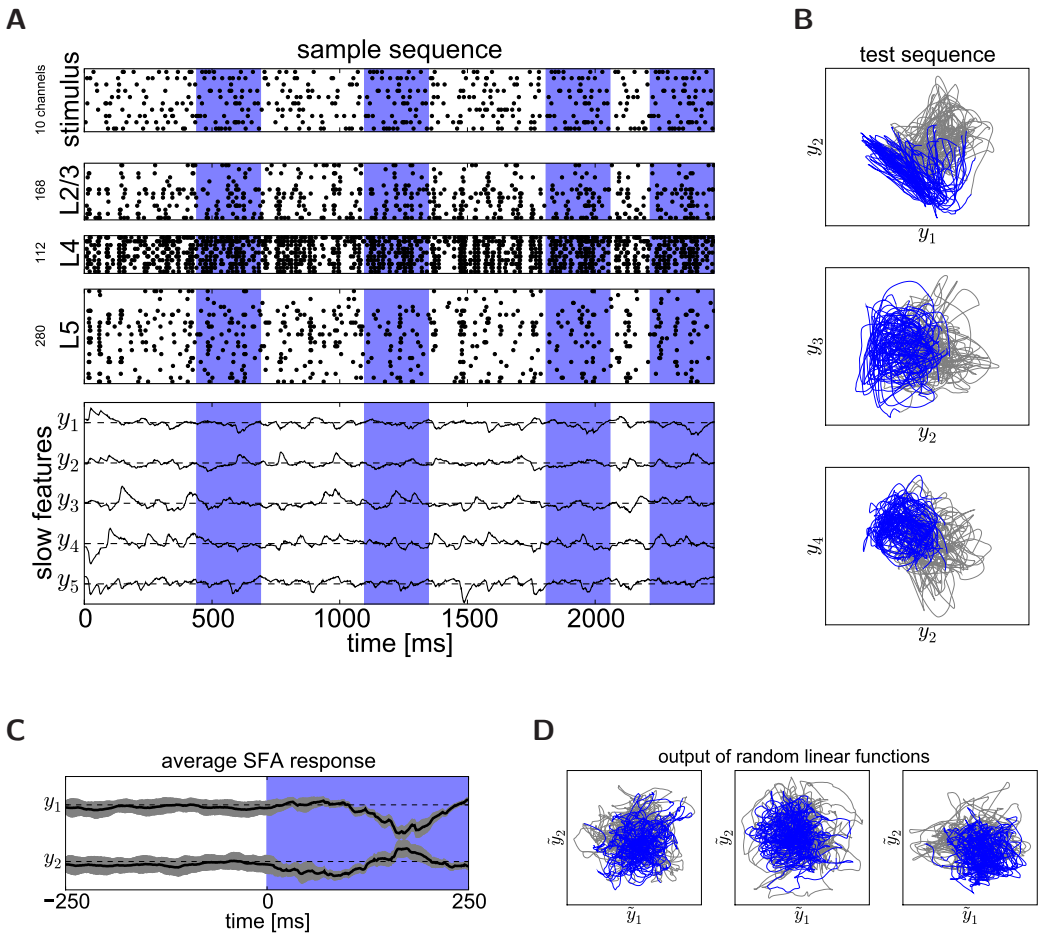

Figure 3.7: (see next page for Figure caption)

to the circuit trajectories is the harder task because of the variability of the response to repeated presentations of the same pattern and because of temporal integration: The circuit integrates input over time making the response during a pattern dependent on the noise input immediately before the start of the pattern. Figure 3.7C shows these two effects. The standard deviation during the noise input is due to different stimulus spike trains which are drawn anew each time. On the other hand, the variability during the pattern presentations results from the inherent noise of the network, i.e., from different responses to the same stimulus. Figure 3.7C shows that the standard deviation during patterns is smaller than during noise. However, at the start of the pattern it does not decrease immediately, but gradually, due to temporal integration. That means that even though the average SFA response becomes different from zero just after the pattern onset, the output still depends on the previous noise input. Figure 3.7C suggests that this forgetting time of the circuit, the time after which the output of the laminar circuit does not depend on the noise any more, is at least 50ms.

Figure 3.7: **Unsupervised learning of the detection of spike patterns.** (**A**) From top to bottom: sample stimulus sequence, response spike trains of the network, and slowest features. The stimulus consists of 10 channels and is defined by repetitions of a fixed spike pattern (blue shaded regions) which are embedded into random Poisson input of the same rate. The pattern has a length of 250ms and is made up by Poisson spike trains of rate 20Hz. The period between two patterns is drawn uniformly between 100ms and 500ms. The response spike trains of the laminar circuit of (Häusler and Maass, 2007) are shown separated into layers 2/3, 4, and 5. The numbers of neurons in the layers are indicated on the left, but only the response of every 12th neuron is plotted. Shown are the 5 slowest features, y_1 to y_5, for the network response shown above. The dashed lines indicate values of 0. (**B**) Phase plots of low-pass filtered versions (leaky integration, $\tau = 100$ms) of individual slow features in response to a test sequence of 50 embedded patterns plotted against each other (blue: traces during the pattern, gray: during random Poisson input). Note that equal increments in x- and y-direction have the same length, i.e., a circle is circular. (**C**) Average response of the two slowest features, y_1 and y_2, during the 250ms spike pattern (blue) and a preceding 250ms noise period (white). Note that the spike pattern is fixed, but the noise is drawn anew each time. The average was taken over 50 pattern repetitions not used for training, as those in **B**. The dashed line denotes the value 0; the shaded area indicates the standard deviation across these 50 repetitions. (**D**) Phase plots of two features $\tilde{y}_1$ and $\tilde{y}_2$ obtained from three randomly chosen orthogonal projections (compare with the top panel in **B**).

3.3.2 Recognizing isolated spoken digits

In the second experiment we tested whether SFA is able to discriminate two classes of trajectories as described in section 3.2.3. We performed a speech recognition task using the dataset considered originally in (Hopfield and Brody, 2000, 2001) and later in the context of biological circuits in (Maass et al., 2002, 2004a) as well as in (Verstraeten et al., 2005) and in (Legenstein et al., 2008). This isolated spoken digits dataset consists of the audio signals recorded from 5 speakers pronouncing the digits "zero", "one", ..., "nine" in ten different utterances (trials) each. We preprocessed the raw audio files with a model of the cochlea (Lyon, 1982) and converted the resulting analog cochleagrams into spike trains that serve as input to our microcircuit model (see Appendix C.2.3.2 for details). This biologically realistic preprocessing is computationally more expensive than the original encoding used in (Hopfield and Brody, 2000), but it has been shown that it can drastically improve the performance of a circuit for a specific speech recognition task (Verstraeten et al., 2005). Figure 3.8A shows sample cochleagrams, stimulus spike trains, and response spike trains for two utterances of digits "one" and "two" by the same speaker.

First, we tried to discriminate between trajectories in response to inputs corresponding to utterances of digits "one" and "two", of a single speaker (speaker 2, as shown in Figure 3.8). We split the 20 available samples (2 digits $\times$ 10 utterances) into 14 training and 6 test samples (i.e., three utterances of each digit is kept for testing). To produce an input to SFA, we generated from these 14 training samples a random sequence of 100 input patterns, recorded for each pattern the response of the circuit, and concatenated the resulting trajectories in time. Note that the same pattern is presented many times. Here we did not switch the classes of two suc-

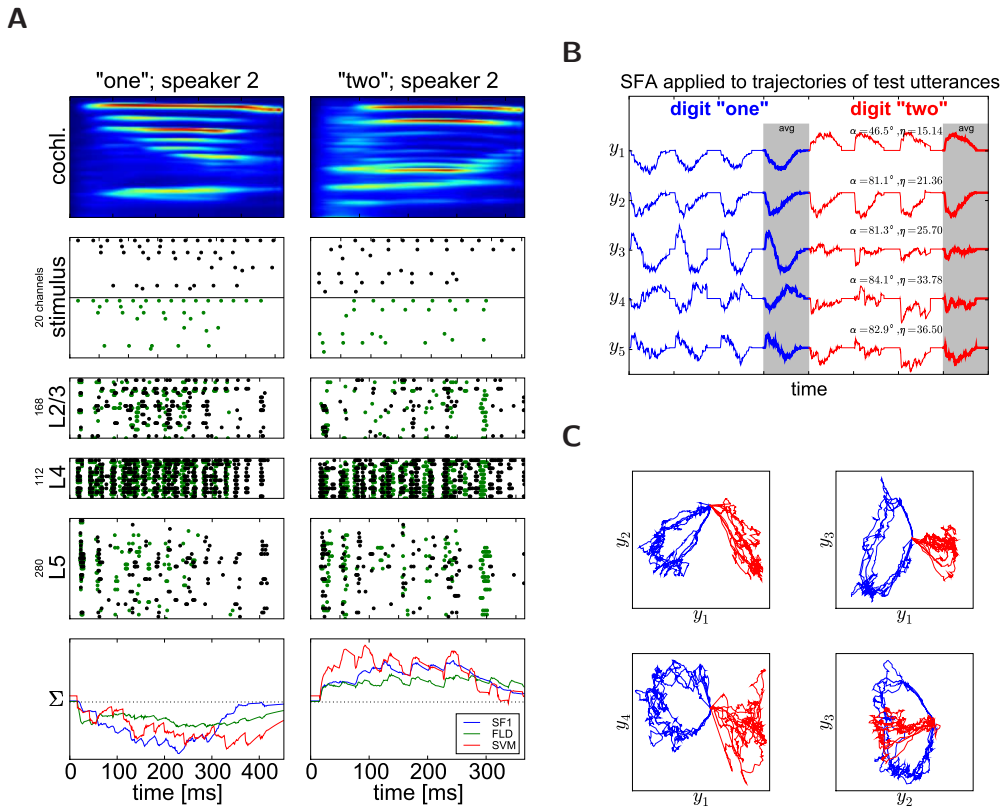

Figure 3.8: (see next page for Figure caption)

cessive trajectories with a certain probability because, as explained in the previous section, for long trajectories the SFA response is independent of this switching probability. Rather, we trained linear SFA readouts on a completely random trajectory sequence.

We then trained linear SFA readouts on the 560-dimensional circuit trajectories, defined as the low-pass filtered spike trains of the spike response of all 560 neurons of the circuit. All responses were recorded for the same amount of time such that all trajectories had the same length; after the circuit activity had stopped, the trajectories descended back to zero. Once there is zero (or noise) input between trajectories the result of SFA becomes independent of the temporal order of the trajectories because only adjacent time steps play a role. However, according to section 3.2.3 this is anyway the case for sufficiently long trajectories. Note that the network responses for repeated presentations of the same stimulus were different due to the inherent noise in the network that was used to model the background synaptic activity *in vivo* (see Appendix C.2.3.2).

Figure 3.8B shows the 5 slowest features, y_1 to y_5, ordered by decreasing slowness in response to the trajectories corresponding to the three remaining test utterances for each class, digit "one" and digit "two". As a measure of slowness we used the

Figure 3.8: **SFA applied to unsupervised digit recognition for a single speaker.** (**A**) From top to bottom: sample cochleagrams, input spike trains, response spike trains of the network, and traces of different linear readouts. Each cochleagram has 86 channels with analog values between 0 and 1 (red, near 1; blue, near 0). Stimulus spike trains are shown for two different utterances of the given digit (black and green; the black spike times correspond to the cochleagram shown above). The response spike trains of the laminar circuit from (Häusler and Maass, 2007) are shown separated into layers 2/3, 4, and 5. The numbers of neurons in the layers are 168, 112, and 280, respectively, but only subsets of these neurons are plotted (14, 10, 24). The responses to the two stimulus spike trains in the panel above are shown superimposed with the corresponding color. Each readout trace corresponds to a weighted sum (Σ) of network states of the black responses in the panel above. The trace of the slowest feature ("SF1", blue line; see **B**) is compared to traces of readouts trained by FLD (green line) and SVM with linear kernel (red line) to discriminate at any time between the network states of the two classes. All weight vectors are normalized to length 1. The dashed line denotes the threshold of the respective linear classifier. (**B**) Response of the 5 slowest features y_1 to y_5 of the previously learned SFA in response to trajectories of the three test utterances of each class not used for training (blue, class 1; red, class 2). The slowness index η (3.26) is calculated from these output signals. The angle α denotes the deviation of the projection direction of the respective feature from the direction found by FLD. The thick curves in the shaded area display the mean SFA responses over all three test trajectories for each class. (**C**) Phase plots of individual slow features plotted against each other (thin lines: individual responses, thick lines: mean response over all test trajectories). Note that equal increments in x- and y-direction have the same length, i.e., a circle is circular.

index η of a signal $y(t)$ defined in (Wiskott and Sejnowski, 2002),

$$\eta(y) := \frac{T}{2\pi}\sqrt{\Delta(y)}. \tag{3.26}$$

This is a slightly different measure than (3.1), and denotes the number of oscillations of a sine wave with the same Δ-value. We found that the two slowest features, y_1 and y_2, responded with shapes similar to half sine waves during the presence of a trajectory (each 500ms a trajectory starts and lasts for several 100ms), which is in fact the slowest possible response under the unit variance constraint. Higher order features partly consisted of full sine wave responses, which are the slowest possible responses under the additional constraint to be decorrelated to previous slow features.

In this example already the slowest feature y_1 extracts the class of the input patterns almost perfectly: it responds with positive values for trajectories in response to utterances of digit "two" and with negative values for utterances of digit "one", and generalizes this behavior to unseen test examples. As a measure for the discriminative capability of a specific SFA response, i.e., its quality as a possible classifier, we measured the angle between the projection direction corresponding to this slow feature and the direction of the FLD. Since each slow feature as well as the weight vector that specifies the projection direction of the FLD is only determined up to the sign, we only report the smaller value. These angular values therefore vary between 0° and 90°. It can be seen in Figure 3.8B that the slowest feature y_1 is closest to the FLD. Hence, according to (3.25), this constitutes an example where

the separation between classes dominates, but is already significantly influenced by the temporal correlations of the circuit trajectories.

We call this property of the extracted features, to respond differently for different stimulus classes, the *What*-information (Wiskott and Sejnowski, 2002). The second slowest feature y_2, on the other hand, responds with half sine waves whose sign is independent of the pattern identity. One can say that, in principle, y_2 encodes simply the presence of a circuit response. This is a typical example of a representation of *Where*-information (Wiskott and Sejnowski, 2002), i.e., the "pattern location" regardless of the identity of the pattern. Full sine wave responses would further encode the position within the trajectory. The other slow features y_3 to y_5 do not extract either *What*- or *Where*-information explicitly, but rather a mixed version of both. For repeated runs of the same experiment with different training utterances the explicit *What*- and *Where*-information of y_1 and y_2 are reliably extracted, but the exact shape of the higher order features might differ depending on the particular training utterances.

Figure 3.8C shows phase plots of these slow features shown in Figure 3.8B plotted against each other. In theory, in the phase plot of two features encoding *What*-information the responses should form straight lines from the origin in a pattern-specific angle. In the three plots involving feature y_1 it can be seen that these response directions are distinct for different pattern classes. On the other hand, phase plots of two features encoding *Where*-information ideally form loops in the phase space, independent of the identity of the pattern, where each point on this loop corresponds to a position in the trajectory. This can only be seen to some extent in the plot y_2 vs y_3, but not explicitly because in this example no two features encode *Where*-information alone. Similar responses have been theoretically predicted in (Wiskott, 2003) and found in simulations of a hierarchical (nonlinear) SFA network trained with a sequence of one-dimensional trajectories (Wiskott and Sejnowski, 2002). Furthermore, we found that the response vector $\mathbf{r}(t) := (y_1(t), \ldots, y_5(t))$, which is composed of the values of all 5 slowest features at a particular point in time, clusters at different directions for different classes. The average angle between two response vectors from different classes is around 90 degrees throughout the duration of a trajectory. This effect arises from the decorrelation constraint and is also a theoretical result of (Wiskott, 2003).

Note that the information extracted by SFA about the identity of the stimulus is provided not only at the end of a specific trajectory, but is made available right from the start. After sufficient training, the slowest feature y_1 in Figure 3.8B responds with positive or negative values indicating the stimulus class during the whole duration of the network trajectory[8]. This supports the aforementioned idea of "anytime computing". Moreover, as a measure for the performance of SFA we can train a linear classifier on the extracted features, i.e., at each point in time the response vector $\mathbf{r}(t)$, composed of the values of the 5 slowest features at that time, and labelled with the class of the corresponding trajectory, serves as one data point for the classification. The performance that a particular classifier is able to

[8]Since the optimal SFA response is not a piecewise constant curve, but a sequence of half sine waves, an even better discriminator would be the direction of the response vector $\mathbf{r}(t)$ which theoretically stays constant throughout a trajectory (Wiskott, 2003).

achieve can be viewed as a lower bound for the information that the extracted slow features convey about the trajectory class. Applied to the features of Figure 3.8B, sampled every 1ms, an SVM with linear kernel achieves a classification performance of 98% (evaluated by 10-fold cross validation). Note again that this is an "anytime" classification, since samples during the whole duration of the trajectories are taken into account.

The bottom panel of Figure 3.8A shows readout traces of three different linear discriminators applied to specific test trajectories, one from each class. Each point on a trace represents a weighted sum of the network states at a particular time, just before the threshold operation of the corresponding linear classifier. That is, a value above (below) zero means that the state at that time is classified to belong to class 2 (class 1) by this particular linear discriminator. Here, we interpret the slowest feature extracted, y_1 from Figure 3.8B, as a linear discriminator with this particular weight vector and the average over the training time series as the discrimination threshold. We compare the trace of this "SFA classifier" to traces of linear readouts trained as Fisher's discriminant and support vector machine (SVM) (Schölkopf and Smola, 2002) to discriminate between the network states of trajectories of different classes[9]. Both FLD and SVM are trained on the same input as SFA, which consists of the network states sampled with $\Delta t = 1$ms of 100 trajectories chosen randomly as described above (but, of course without the information about the temporal sequence of states). The discrimination threshold for both SFA and FLD was chosen as the average over all training points. It can be seen that in this case the slowest feature, which has been learned in an unsupervised manner, is able to achieve a perfect separation, comparable to those of the supervised methods of FLD and SVM. That is, if we interpret the weight vector of this slowest feature as the weight vector of a linear discriminator, this classifier achieves a performance of almost 100% on deciding which class of input stimuli has caused these unseen network state trajectories, even in an "anytime" manner, i.e., during the whole duration of the trajectories.

Figure 3.9A shows the responses of SFA trained on a sequence of 500 trajectories corresponding to utterances of digits "one" and "two" of all 5 speakers. From the 100 available samples (2 digits $\times$ 5 speakers $\times$ 10 utterances) we have used 70 for training and kept the remaining 30 for testing. The response of the learned SFA to trajectories in response to three of these testing utterances for each of the two classes, as well as the mean SFA response over all 30 test utterances of each class, is shown in Figure 3.9A. It can be seen that, qualitatively, the performance decreases compared to the case where only a single speaker is used (see Figure 3.8B). No single feature extracts the class information alone, but significant *What*-information is still represented: First, the slowest feature y_1 responds more strongly to trajectories corresponding to samples with digit "one". Second, feature y_3 responds with negative values only for trajectories in response to digit "two", whereas for those of digit "one" it consistently has an initial positive response. Again feature y_1 has the smallest angular distance to the FLD direction, even if it is larger than

[9]Note that the absolute scale of different readout traces relative to each other is arbitrary since only the direction of the weight vectors are relevant. In this presentation all three weight vectors are normalized to length 1, in order to be comparable to each other.

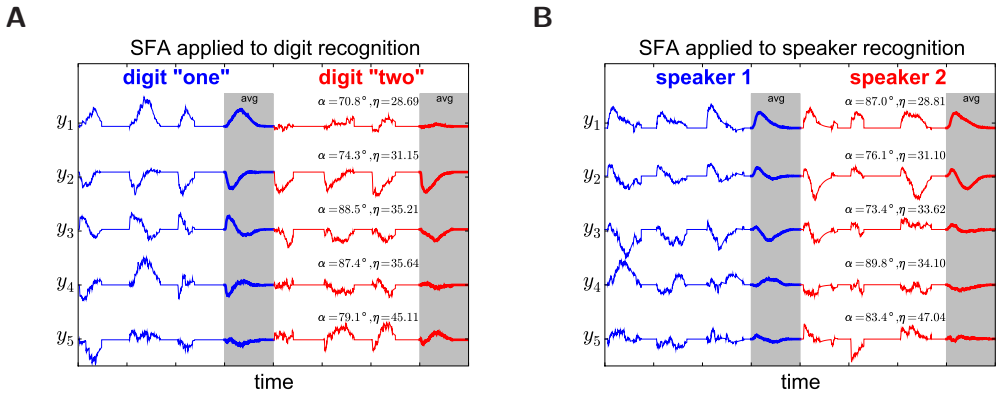

Figure 3.9: **SFA applied to unsupervised speaker-independent digit recognition and digit-independent speaker recognition.** Both panels show the response of the 5 slowest features y_1 to y_5 of the previously learned SFA in response to trajectories of three test utterances of each class not used for training. Trajectories are padded with zeros such that each trajectory has the same length. The slowness index η (3.26) is calculated from these output signals. The angle α denotes the deviation of the projection direction of the respective feature from the direction found by FLD. The thick curves in the shaded area display the average SFA responses over all available test trajectories for each class. (**A**) SFA applied to speaker-independent digit recognition. Shown are the responses for three random test trajectories of digit "one" (blue) and digit "two" (red) from three different speakers as well as the average SFA response over all 30 available test trajectories. (**B**) SFA applied to digit-independent speaker recognition. Shown are the responses for three random test trajectories of speaker 1 (blue) and speaker 2 (red) from three different digits as well as the average SFA response over all 60 available test trajectories.

in Figure 3.8B.

Similarly, we can apply SFA to a sequence of trajectories in response to utterances of speakers 1 and 2 (but now with all 10 digits) try to extract information about the speaker feature, independent of the spoken digit. Now there are 200 available samples (10 digits $\times$ 2 speakers $\times$ 10 utterances), where we have used 140 for training and kept the remaining 60 for testing. Figure 3.9B shows the responses of the learned SFA to 3 trajectories of these test utterances, as well as the average SFA response over all 60 test trajectories. Due to the increased number of different samples for each class (for each speaker there are now 10 different digits) this task is more difficult than the speaker-independent digit recognition. No single slow feature extracts *What*-information alone; the closest feature to the FLD is feature y_3. To some extent also y_4 extracts discriminative information about the stimulus.

In these experiments, the separation between the classes (expressed by the first term in (3.25)) obviously decreases compared to the single-speaker case. In such a situation where the distance between the class means is very small, the tendency to extract the trajectory class itself as a slow feature becomes negligible. In that case the theory predicts that SFA tries to distinguish each individual trajectory due to the decorrelation constraint, and clusters similar trajectories because of the finite

nonlinear expansion	# dimensions	classifier performance
none (stimulus)	20 (10)	75%
quadratic	65	81%
cubic	285	83%
laminar circuit	560 (100)	88%

Table 3.1: **Performance values of a linear classifier trained on the slow features in response to different nonlinear expansions of the input, for the speaker recognition experiment in Figure 3.9B.** The nonlinearity implicitly provided the laminar circuit is compared to a quadratic and cubic expansion of the stimulus, as well as to the naked stimulus. The second column gives the dimensionality of the state space provided by the respective nonlinear projection. The numbers in brackets denote the effective dimensions used to train SFA, after PCA is applied (see Appendix C.2.3.3). The quadratic (cubic) kernel contains all monomials up to degree 2 (3) of the 10 effective stimulus dimensions (Wiskott and Sejnowski, 2002). Performance values are evaluated by 10-fold cross validation.

(linear) function space. It can be seen in Figure 3.8 that higher-order features start to discriminate between different samples of the same class. This demonstrates that multiple SFA responses are important and collectively convey discriminative information about the class of the trajectory currently being presented, and that in these examples one should view SFA as a powerful preprocessing stage for a subsequent classification, rather than a classifier itself.

It is important to note that the different classification results in Figures 3.9A and 3.9B are not obtained due to a different temporal order of the trajectories within the training input (i.e., whether the speaker is varying more slowly than the digit, or vice versa), but due to the use of a different training set of trajectories. The result of SFA does not depend on the temporal order of the trajectories within the training input because of the intermittent zero phases, and is therefore completely determined by the training set of trajectories.

The performance of a linear classifier trained on the 5 slowest features in response to all available test trajectories to predict the class label of the stimulus is 90% for the speaker-independent digit recognition (Figure 3.9A) and slightly lower (88%) for the digit-independent speaker recognition (Figure 3.9B). If linear SFA is applied directly to the 20-dimensional trajectories obtained by low-pass filtering the stimulus spike trains directly, the same classifier achieves a performance of about 75%. This indicates that the circuit provides a useful nonlinear combination of input components. Table 3.1 compares these performance values to different nonlinear expansions of the stimulus for this experiment. It can be seen that the laminar circuit yields a better performance than a cubic kernel, even though the number of dimensions already have the same order of magnitude. Other than the quadratic and cubic expansion, which are static, the circuit additionally provides a temporal integration of the stimulus which might provide a significant performance improvement in this case.

Note again that these are performance values for an unsupervised "anytime" speech recognition task. A comparable performance has been achieved in (Maass

et al., 2004a) on a different task (digit "one" against all other digits) on the encoding by (Hopfield and Brody, 2000) by training the readout weights with a linear SVM. The performance values reported in (Verstraeten et al., 2005) are not for "anytime" speech recognition in the sense that snapshots across different time points of network trajectories are used for training the readout, but a majority vote across different classifiers trained at different time points is used to predict the currently spoken digit. If the decision about which stimulus class has been presented should not be made "anytime", but only at the end of each stimulus/trajectory, almost perfect performance can be achieved by integrating the slow features during the duration of a trajectory, i.e., by accumulating evidence for or against a given speaker or digit.

Finally, we note that the qualitative performance of SFA (i.e., how "good" the features look, or in which order the features are extracted) depends on the smoothness of the trajectories that are used for training. The circuit model of (Häusler and Maass, 2007) typically shows a bursting behavior, which is mostly due to the short-term dynamics of synapses. Thus the performance of SFA can even be improved by using a circuit model that generates smoother trajectories of network states. Also, we obtain similar results if we apply SFA directly on a sequence of the high-dimensional analog cochleagrams shown in Figure 3.8A.

3.4 Discussion

3.4.1 SFA as a principle for neural computation

We have shown in this Chapter that slow feature analysis (SFA) can in principle be used for learning *unsupervised* (or implicitly supervised) linear discrimination. SFA (Wiskott and Sejnowski, 2002) belongs to a family of algorithms for learning invariant representations from *temporal* input sequences, which maximize the "slowness" of their output signals (e.g., Földiak, 1991; Mitchison, 1991; Becker and Hinton, 1992; Stone and Bray, 1995). This objective is based on the assumption that signals that encode invariant representations, such as the location or identity of an object in the visual field, vary on a much slower time scale than raw sensory signals, such as the intensity of the visual input at a single fixed point on the retina, for example. Therefore, the extraction of slow features of the quickly varying input signal is likely to yield invariant properties of this input signal. The unique aspect about SFA is its appealing formulation as an eigenvalue problem in the covariance matrices of the (possibly nonlinearly expanded) multi-dimensional input signal.

This formulation has allowed us to establish a relationship between this unsupervised learning rule and a powerful supervised method for classification, Fisher's linear discriminant (FLD), which can be expressed as a similar eigenvalue problem. In particular, we have demonstrated that by converting the input to a classification problem (two labeled point sets) into an unlabeled time series in a special way, SFA is able to closely resemble the result of FLD on this classification problem. More precisely, if two consecutive points in the time series are likely to be chosen from the same class (i.e., the switching probability p between the classes is low), both methods yield similar projection directions, which can be interpreted as hypotheses of linear discriminators (i.e., separating hyperplanes). Due to this tendency that

temporally contiguous points are from the same class SFA is able to learn to become invariant to different points within a class, but to respond differently for points from different classes, i.e., to extract the class as a slowly varying feature.

In this Chapter we have basically considered three cases of application of SFA for pattern recognition: (i) point discrimination, (ii) trajectory discrimination with different means, and (iii) trajectory discrimination with identical means. In case (i), the point discrimination, the class membership is implicitly encoded in the temporal sequence of samples that serves as input to SFA. The optimal response is a piecewise constant function during periods where points from the same class are presented, and for a linear function, converges to the result of FLD on the original classification problem. Regarding case (ii), the trajectory discrimination with different means, we have analyzed how the SFA objective changes if it is applied to a time series that consists of a sequence of such trajectories of training examples instead of individual points that are independently chosen at each time step. More precisely, we have considered a trajectory classification problem, which consists of sets of point sequences rather than sets of individual points. We generated a time series from this classification problem by randomly choosing trajectories from these two sets and by concatenating them into a single sequence. We found that for such a sequence of sufficiently long trajectories the result of SFA becomes independent of the class switching probability between two successive trajectories, thus of the temporal order of the trajectories within the time series. Applied to such a time series, the optimization problem of SFA can be viewed as a composition of two effects: the tendency to extract the trajectory class as a slow feature and the tendency to produce a smooth response during individual trajectories. The first effect can be described by the scatter matrices of the FLD, whereas the second effect depends on the temporal correlations (with time lag 1) of the trajectories.

Case (iii) occurs when the class means are so close together that they are almost identical. In this case the effect of the FLD vanishes. If the trajectories are interleaved with zero (or noise) input the optimal solution to SFA would be to respond with a half sine wave to each trajectory. For that case, (Wiskott and Sejnowski, 2002; Wiskott, 2003) explain the emergence of discriminative information in the SFA responses by the decorrelation constraint: a feature that responds with different amplitudes for different patterns also varies slowly and can still be decorrelated to other features that exhibit the same response for each pattern. Thus, with an infinite function space SFA always produces a feature that responds with a different amplitude for each individual pattern. If the available function space is limited (e.g., linear, as in our case) SFA might cluster similar trajectories, e.g., those that belong to the same class, by responding to them with similar amplitude.

In the context of biologically realistic neural circuits this ability of an unsupervised learning mechanism is of particular interest, because it could enable readout neurons, which typically receive inputs from a large number of presynaptic neurons of the circuit, to extract from the trajectory of network states information about the stimulus that has caused this particular sequence of states – without any "teacher" or reward. In previous simulation studies of neural circuit models, so far training of readouts of biological microcircuits has mostly been performed in a supervised manner (Maass et al., 2002, 2004a; Legenstein et al., 2005) or in a reward-based

trial-and-error setting (Legenstein et al., 2008).

We have tested the potential biological relevance of this learning principle in computer simulations of a quite realistic model of a cortical microcircuits (Häusler and Maass, 2007). More precisely, we have tested whether SFA would enable projection or "readout" neurons to learn without supervision to detect and discriminate salient input streams to the microcircuit. These readouts were modelled as linear neurons, i.e., we have neither used a particular nonlinear expansion (Wiskott and Sejnowski, 2002), which would have likely suffered from the curse of dimensionality when applied to these high-dimensional trajectories, nor an explicit kernel (Bray and Martinez, 2003). Rather, we have taken advantage of the kernel property of the circuit itself, which provides intrinsic nonlinear combinations of input components by its recurrent connections, and thereby boosts the expressive power of a subsequent linear readout.

In particular, we have shown that SFA is able to detect a repeating spike pattern within a continuous stream of Poisson input with the same firing statistics in an unsupervised manner. Furthermore, we demonstrated that the recognition of isolated spoken digits is possible using a biologically realistic preprocessing for audio samples. SFA was able to almost perfectly discriminate between two digits of a single speaker, and to a lesser extent also to extract information about the spoken digit independent of the speaker as well as the speaker independent of the spoken digit.

The laminar circuit transforms the input spike trains in three different ways. First, it provides a nonlinear expansion of the input by projecting it into a higher dimensional space through its recurrent connections. We have shown in one of the speech discrimination tasks that the circuit significantly improves the performance of a subsequent linear SFA readout compared to the case where this readout is directly applied to the stimulus spike trains. Moreover, the circuit performs better than a static quadratic or cubic expansion of the stimulus (see Table 3.1). The second effect of the circuit is to provide temporal integration. While this may be benefitial in the spoken digits tasks, it certainly decreases the performance in the pattern detection task because it makes the response of the circuit at the beginning of a pattern depend on the noise input immediately before (see Figure 3.7C). The third effect is the inherent noise of the network, i.e., its property to respond differently each time the same stimulus is presented. This noise models the background synaptic input *in vivo* (Destexhe et al., 2001). SFA should perform better if this intrinsic noise is low.

We find that the response of the learned SFA readouts to a sequence of test trajectories contains both *What*- and *Where*-information, i.e., they encode the class of the trajectory currently presented (pattern identity) as well as the current position within a trajectory (location within a pattern). This is in agreement with the objective of SFA, because both the location and identity vary on a slower time scale than the raw sequence of network states. The extracted features tend to be sections of sine waves, which are the slowest possible responses under the constraints of unit variance and decorrelation. Features encoding *Where*-information usually detect the presence of the trajectories (and encode the current position within the trajectory) independent of their identity and respond with similar shapes to each

trajectory. Such information is very useful for neural systems, since it allows them to keep track of time relative to a stimulus onset or the initiation of a motor response (Buonomano and Mauk, 1994; Buonomano and Maass, 2009). The fact that such timing information becomes automatically available through unsupervised SFA could in fact point to a general advantage of coding and computing with trajectories of firing states, rather than with single firing states (as many classical theories of neural computation propose). Obviously the ability to keep track of time on the time scale of a few hundred ms is essential for biological organisms, e.g., for motor control. In contrast, features encoding *What*-information discriminate between different types of trajectories and respond differently for different classes of trajectories. The response vector defined by the slowest features at a particular point in time takes on specific directions for each trajectory class; we have found that the average response vectors of different classes are around 90 degrees apart. These properties of SFA have been theoretically predicted in a thorough analysis of this algorithm (Wiskott, 2003). In (Wiskott and Sejnowski, 2002), they have been also found in computer simulations, where a hierarchical SFA network has been trained with a sequence of short one-dimensional trajectories. There, the particular organization of sequential quadratic SFA stages provides the nonlinear function space, from which the function is chosen that generates the slowest possible output from the input signal. In our case this function space is implicitly given by the nonlinearity of the circuit.

In contrast to the results for classification problems on point sets, due to the temporal structure of trajectories a single SFA readout of a cortical microcircuit might not extract the class of network trajectories explicitly, but usually a mixture of both *What*- and *Where*-information. This is what we had expected from our theoretical analysis, which suggested that there is a trade-off between the tendency to separate different classes and the tendency to respond as smoothly as possible during individual trajectories. Moreover, as the distance between the class means decreases, the separation tendency becomes negligible, and SFA tries to distinguish all individual trajectories. However, the slowest features span a subspace where the trajectories are nicely separated, thereby rendering SFA a powerful preprocessing stage by improving the computational performance of a subsequent classification. Furthermore, the results show that SFA readouts are able to distinguish between different stimulus classes in an "anytime" manner, i.e., they provide the correct classification already before the trajectory has ended. This makes the information about the stimulus available to later processing or decision making stages not only after a trajectory has settled into an attractor, but already while the stimulus is still being presented.

In these circuit simulations, SFA responds with amplitudes of different sign to patterns of different classes, and even generalizes this behavior to unseen test examples. We argue that the function space that is implicitly provided by the cortical microcircuit together with the linear SFA readouts might just have the property that different trajectories yield the same responses if they are similar enough. More precisely, it might correspond to an imperfect kernel that maps similar input patterns (patterns that are likely to be from the same class) into similar trajectories, and sufficiently distinct input patterns to trajectories that are

significantly separated. Previous studies (e.g., Legenstein and Maass, 2007) suggest that if such circuits operate in a regime called *edge of chaos*, they might have this desired property.

Furthermore, our theory predicts and our experiments show that the ability of SFA to discriminate between different classes of trajectories is strongly influenced by the temporal correlations of the trajectories, as explained by the temporal covariance matrix with time lag 1. It would be interesting to investigate the effect of different magnitudes of these correlations, e.g., by comparing the effect of different sampling frequencies (we use a quite short sampling time in our examples).

3.4.2 Relation to preceding work

Slow feature analysis has already been applied for unsupervised pattern recognition in (Berkes, 2005b, 2006), where SFA has been used to discriminate between handwritten digits. There the SFA objective is reformulated to optimize slowness for time series consisting of just two patterns, averaged over all possible pairs of patterns. The idea is to search for functions that respond similarly to patterns of the same class, and therefore ignore the transformation between the individual patterns. The optimization (3.1) in (Berkes, 2006) is performed over the set of time derivatives of all possible pairs of samples of a class,

$$\min \quad \Delta(y_j) = a \cdot \sum_{c=1}^{C} \sum_{\substack{k,l=1 \\ k<l}}^{N_c} \left(g_j(\mathbf{x}_k^c) - g_j(\mathbf{x}_l^c)\right)^2, \tag{3.27}$$

under the constraints of zero mean, unit variance, and decorrelation, where C is the number of classes, N_c is the number of samples of class c, $\mathbf{x}_k^c$ is the k-th sample of class c, and a is a normalization constant dividing by the number of all possible pairs. Obviously, the functions g_j that minimize (3.27) are ones which are constant for all patterns belonging to the same class, in which case the objective function is zero. As a consequence, patterns from the same class will cluster in the feature space formed by the output signals of the $(C-1)$ slowest functions g_j, where classification can be performed using simple techniques (Berkes, 2005b, 2006).

One problem with this approach is that it is often computationally intractable to consider all pairs of patterns, since the number of pairs grows very fast with the number of patterns. Furthermore, it might be implausible to have access to such an artificial time series, e.g., from the perspective of a readout of a cortical microcircuit which receives input on-the-fly. We take a different approach and apply the standard SFA algorithm to a time series consisting of randomly selected patterns of the classification problem, where we switch the class of the current pattern at each time step with a certain probability. We have found that if this switching probability p is low SFA extracts features which separate the classes and finds approximately the same subspace as Fisher's linear discriminant. In particular, we have demonstrated the dependence of the deviation on p: as p goes to zero, the weight vector of SFA converges to the weight vector of FLD. Note that with this approach perfect equivalence between SFA and FLD cannot be reached because the time series would have to consist only of transitions within a class, but at the same

time contain patterns from all classes, which is not possible. In this hypothetical case the SFA problem would become equivalent to the reformulated objective in (Berkes, 2005b, 2006). In (Berkes, 2005a) the author applied a nonlinear version of Fisher's discriminant to the same handwritten digits dataset as in (Berkes, 2005b, 2006) (using a fixed polynomial expansion of the input as a kernel) and achieved a similar result, but, however, no relationship between the two methods was shown.

In (Franzius et al., 2008) the authors show that a hierarchical network of quadratic SFA modules can extract the identity of objects from an image sequence presenting these objects at continuously changing positions, sizes, and viewing angles. They create the input sequence in a similar way as we do: after each time step, the object identity is switched with a low probability. The resulting features extracted by SFA contain information about the identity of the object currently shown, as well as the current position, size and rotation angles of the object. However, this information is usually not made explicit in the sense that a single slow feature codes for exactly one "configuration variable" (such as object identity or position), rather, each such variable is distributed over multiple slow features. The original variables, however, can be recovered from the slowest features using linear regression or simple classifiers with high accuracy. The tendency for this "linear mixing" of information increases as the input sequence gets more and more complex (i.e., contains more transformations of the same object). We also find this effect in our experiments: in the experiment where we discriminated between spoken digits of a single speaker (Figure 3.8) the slowest feature extracted the class information explicitly, whereas in the experiment where more speakers were used (Figure 3.9A) this information was distributed over multiple features. In principle, one can view the nonlinear expansion of the image sequence that belongs to a single object presentation (i.e., between two object switching events) as a particular trajectory in response to this object. Different trajectories for the same object vary in the specific sequence of poses of that object during a particular presentation phase. In this sense, SFA is trained on a sequence of trajectories, each resulting from a specific presentation of a particular object. According to (Franzius et al., 2008) the classifier performance for extracting the object identity is maximized if all other variables are made very fast. This is in agreement with our theory because faster configuration variables produce weaker temporal correlations of these image trajectories. This means that SFA more closely approximates the result of FLD on these images.

This work in (Franzius et al., 2008) offers one explanation how the visual system learns invariant object recognition from the temporal statistics of the input stimuli: images that occur in immediate succession tend to belong to the same object. In fact, a considerable amount of work has been done that investigates temporal slowness as a computational principle in the visual system. In (Berkes and Wiskott, 2003) quadratic SFA (i.e., linear SFA in the expanded input of all polynomials of degree 2 of the original input dimensions) has been applied to natural image sequences and the learned quadratic forms have been interpreted as receptive fields (Berkes and Wiskott, 2006). These resulting receptive fields resemble many properties of complex cells, such as their Gabor-like shape, shift invariance, or direction selectivity. Furthermore, when presented with a visual input sequence that is generated by the movement of a simulated rat in a virtual environment, SFA has been shown

to reproduce the spatial firing patterns of place cells, head-direction cells, spatial-view cells, and grid cells (Franzius et al., 2007a). Depending on the movement statistics of this simulated rat different types of invariances are learned (e.g., the head direction independent of the current position in the environment). To obtain the final response characteristics of these cell types, however, an additional sparse coding stage has been incorporated (Franzius et al., 2007b), which extracts the representations of single cells from the more distributed representations resulting from SFA. Using a slightly different slowness objective, similar invariance properties of the visual system have been found in (Einhäuser et al., 2002; Wyss et al., 2006).

In this work we have trained the readouts of our cortical microcircuit with the standard batch algorithm of SFA. This has the advantage that there are no parameters which have to be tuned for a specific problem. Since SFA is based on an eigenvalue problem it finds the solution in a single iteration and has no convergence problems (e.g., to be trapped in local minima). In biological systems, however, processing has to be performed on-the-fly, and therefore learning rules that optimize temporal stability in an online manner are of particular interest. Several computational models exist that are based on this slowness principle and that show how invariances in the visual system can be learned through a variety of Hebbian-like learning rules (Földiak, 1991; Wallis and Rolls, 1997; Wyss et al., 2006; Masquelier and Thorpe, 2007). In this Chapter we do not propose a biologically realistic learning rule, rather, we investigate the properties of one well-known algorithm, slow feature analysis, out of this family of optimization methods based on the slowness principle and analyze its unsupervised discrimination capabilities. A recent paper demonstrates that this learning rule can in principle be implemented by a spiking neuron with a form of STDP (Sprekeler et al., 2007). Although this result is purely analytical and has yet to be verified in computer simulations, it supports the hypothesis that the objective of slowness is an important ingredient in the unsupervised learning mechanisms of biological systems. In fact, STDP has been successfully applied to robust online unsupervised detection of repeating spatiotemporal spike patterns hidden within spike trains of the same firing statistics (Masquelier et al., 2009). Moreover, it has been shown that spiking neurons equipped with a special form of STDP which receives a global reward signal (Izhikevich, 2007) can learn to discriminate between different trajectories of firing states using reinforcement learning (Legenstein et al., 2008).

SFA is not only inspired by the slowness principle for learning invariances, but might also be motivated by information-theoretic principles, such as the Information Bottleneck (IB) method (Tishby et al., 1999), or Independent Component Analysis (ICA) (Hyvärinen et al., 2001). In (Creutzig and Sprekeler, 2008) a relationship is shown between SFA and the IB method for predictive coding, which optimizes the objective to compress the information of the past into the current state of a system, such that as much information as possible about the future is preserved. In other words, it minimizes $I(\text{past};\text{state}) - \beta I(\text{state};\text{future})$ with some trade-off parameter β. It turns out that for the case of one-time-step prediction and of a linear system with Gaussian noise this problem becomes equivalent to linear SFA. On the other hand, ICA tries to uncover statistically independent signals from an observed linear mixture of these signals. (Blaschke et al., 2006, 2007) show that

for a particular measure of independence, which involves the temporal correlations with a time delay of one time step, ICA becomes formally equivalent to linear SFA. Finally, (Turner and Sahani, 2007) provides a probabilistic interpretation for SFA, where it is assumed that the observed time series $\mathbf{x}$ is generated by a linear mixture of latent variables (the slow features y_i). The mixing matrix $\mathbf{W}$ is recovered by maximizing the likelihood function. This attractive formulation has the advantages that constraints and extensions can be included in the model in a very natural way, and that noise and missing data in the input are handled elegantly by this probabilistic setup. These results establish an interesting connection between the slowness objective and both probability and information theory and further demonstrate the power of the elegant algorithm of linear SFA.

3.4.3 Conclusion

Summarizing, we have established a theoretical basis that explains when slow feature analysis can be expected to have emergent pattern discrimination capabilities. Both our theoretical results and our computer simulations suggest that slow feature analysis – and more generally the concept of slowness or temporal stability – could be a powerful mechanism for extracting temporally stable information from trajectories of network states of biological circuits without supervision, and hence an important ingredient for spatiotemporal processing in cortical networks (Buonomano and Maass, 2009). In particular, it provides a basis for explaining how brains can arrive at stable percepts in spite of continuously changing network states in a completely unsupervised way.

3.5 Acknowledgments

This Chapter is based on the paper *A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction*, written by myself (SK) and my supervisor Wolfgang Maass (WM). The derivation and analysis of the relationship between slow feature analysis and Fisher's linear discriminant was performed by SK. The simulation experiments were designed by SK and WM and conducted by SK. The paper was written by SK, with additional input from WM. We are very grateful to Laurenz Wiskott who provided particularly helpful comments on this paper. Furthermore, we would like to thank Henning Sprekeler and Lars Büsing for stimulating comments and discussions. The paper was written under partial support by the Austrian Science Fund FWF project # S9102-N13 and project # FP6-015879 (FACETS), project # FP7-216593 (SECO) and project # FP7-231267 (ORGANIC) of the European Union.

Chapter 4

Neurons in primary auditory cortex integrate information about past and present stimuli

Contents

This Chapter provides novel experimental evidence for the liquid computing model. The activity of neurons in the primary auditory cortex of awake ferrets is analyzed and it is shown that sequentially arriving stimulus information is integrated over time and superimposed in a non-linear way into the neural responses at one point in time. This Chapter is based on the paper Neurons in primary auditory cortex integrate information about past and present stimuli *by Stefan Klampfl, Stephen V. David, Pingbo Yin, Shihab A. Shamma, and Wolfgang Maass (submitted for publication, 2011). This was a collaboration with the University of Maryland, where all the experimental work was conducted and who provided the data. SVD was also involved in writing the paper and provided additional ideas for the data analysis.*

In order to process the rich temporal structure of their acoustic environment, organisms have to integrate information over time into an appropriate neural response. Previous studies have addressed the modulation of responses of auditory neurons to a current sound depending on the immediate stimulation history, however, it has remained unknown how and where this important computation step is carried out. In this study, we analyzed the temporal integration capabilities of 122 single neurons in primary auditory cortex (A1) of four awake ferrets in response to random tone sequences. We quantified the information contained in the responses about both current and preceding sounds in two ways: by estimating directly the mutual information between stimulus and response, and by training linear classifiers to decode information about the stimulus from the neural response. We found that (i) many neurons conveyed a significant amount of information not only about the

current tone, but also simultaneously about the previous tone, (ii) that the neural response to tone sequences was a non-linear combination of responses to the tones in isolation, and (iii) that, nevertheless, much of the information about current and previous tones could be extracted by linear decoders. These results suggest that A1 neurons perform a generic preprocessing of the temporal structure of auditory stimuli for higher areas. Moreover, this study shows that standard machine learning methods such as classifiers provide a suitable estimate for the actual mutual information, which could lay the ground for a new paradigm of experimental data analysis.

4.1 Introduction

A number of studies in the auditory cortex have demonstrated that neurons respond differently to a certain sound, depending on which sensory events preceded this sound by several 100ms. For example, it has been shown in monkeys (Brosch et al., 1999; Malone et al., 2002; Bartlett and Wang, 2005; Yin et al., 2008), cats (McKenna et al., 1989; Brosch and Schreiner, 2000), and rats (Kilgard and Merzenich, 1999) that the responses to a given tone can change if another tone has been played immediately before. For different configurations of frequency, intensity, and temporal separation the response to this second tone can be facilitated or suppressed. Furthermore, it has been shown that neurons in songbirds respond preferentially to the bird's own song, but weakly to a different sequence of song syllables (Margoliash and Fortune, 1992; Doupe, 1997; Lewicki and Arthur, 1996). Nevertheless, it has remained unclear how the auditory system performs this integration of the incoming, temporally highly structured acoustic information into an appropriate neural response which enables further processing of this information by higher areas.

In recent years a new computational model has been proposed that could explain how this important computation step is carried out. This *liquid computing model* (Maass et al., 2002; Buonomano and Maass, 2009) views cortical circuits as generic preprocessing components that combine sequentially arriving input components, possibly in a non-linear manner, in order to facilitate further information processing by simple readout neurons in higher cortical areas. This model has attracted substantial interest in the computational as well as the experimental neuroscience community, e.g., (Sussillo and Abbott, 2009; Nikolic et al., 2009; Bernacchia et al., 2011). However, so far experimental evidence for this computing model in sensory systems has only been reported for anesthetized cats (Nikolic et al., 2009), and it is not exactly clear how anesthesia affects sensory information processing. One goal of this work is to test the predictions of this model for neural responses in the primary auditory cortex of awake animals. To quantify the temporal integration capability of this auditory system, we analyzed the spike responses of individual A1 neurons of four awake ferrets to random tone sequences and measured the information contained in the neural responses about both current and preceding sounds.

The most principled and rigorous way to analyze the information contained in neural responses is to directly use methods from information theory (Cover and Thomas, 1991). However, it is well-known that the direct estimation of the mutual

information between stimulus and response suffers from a severe bias problem due to the limited number of available trials (Miller, 1955; Panzeri and Treves, 1996). The method proposed in (Panzeri et al., 2007) produces reliable information estimates also for quite limited sample sizes and has been successfully validated in a variety of analysis studies, e.g., in the rat barrel cortex (Arabzadeh et al., 2004, 2006) and in the monkey visual cortex (Montemurro et al., 2008). We used this method to quantify the temporal integration capability of single A1 neurons, by estimating the amount of information they convey at a particular point in time simultaneously about currently and previously played tones.

An alternative method to investigate this temporal integration of information is to take the perspective of a hypothetical neuron that receives input from all simultaneously recorded neurons and to measure how much information about previous stimuli it can read out just from the current response (Buonomano and Maass, 2009). As a first approximation, one can view such readout neurons as linear discriminators because they compute a weighted sum of these inputs and become active once this sum exceeds a certain threshold. The optimal performance that can be obtained by optimizing the weights of this linear discriminator is a measure of the information contained in the neural responses. This method of information analysis has been used in the aforementioned study (Nikolic et al., 2009) to provide evidence for similar temporal integration capabilities in the primary visual cortex. In addition to the direct estimation of mutual information, we applied this method by training state-of-the-art classifiers, Support Vector Machines (SVMs) with linear kernel (Schölkopf and Smola, 2002), on the responses of simultaneously recorded neurons to discriminate between different stimulus conditions.

With this method we also tested whether the neural response provides a non-linear combination of input components, which would be revealed if these linear classifiers achieved a significant performance on a non-linear target function. Moreover, we measured the amount of information both methods were able to extract from the same data and tested whether linear classifiers provided a suitable lower bound on the information measured directly. These findings indicate that neurons already at this early stage of the auditory system provide a generic preprocessing of the complex temporal structure of acoustic stimuli in order to ease the extraction of information by simple readout mechanisms in higher cortical areas, and thus provide substantial experimental evidence for the liquid computing model in the primary auditory cortex of awake animals.

4.2 Materials and Methods

4.2.1 Experimental procedures

Auditory responses were recorded extracellularly from single neurons in primary auditory cortex (A1) of four awake, passively listening ferrets. All experimental procedures conformed to standards of the National Institutes of Health and the University of Maryland Animal Care and Use Committee. Details of the surgical and neurophysiology procedures are described elsewhere (David et al., 2009) and briefly summarized here. Animals were implanted with a stainless steel head post to

permit stable recordings. Single unit activity was recorded from four independently movable high-impedance (2-4 M-Ohm) electrodes in a sound-attenuating chamber. Spiking events were extracted from the continuous signal using principal components analysis and k-means clustering. In total, 122 single A1 neurons were isolated from 23 multi-channel recordings.

Stimuli were presented from digital recordings using custom Matlab software (Mathworks, Natick, MA). Digitally synthesized sounds were transformed to analog (National Instruments, Austin, TX), equalized to achieve flat gain (Rane, Mukilteo, WA), amplified to a calibrated level (Rane, Mukilteo, WA) and attenuated (Hewlett Packard, Palo Alto, CA) to the desired sound level. These signals were presented through an earphone (Etymotics, Elk Grove Village, IL) contralateral to the recording site. Before each experiment, the equalizer was calibrated according to the acoustical properties of the earphone insertion. In each experiment, stimuli were presented at a fixed sound level (65 dB SPL).

4.2.2 Tone sequence stimuli

During each recording random tone sequences were presented as stimuli to the passively listening animal. Individual trials consisted of 100 tones, each of which had a duration of 150ms; thus, the duration of each sequence was 15s. Table 4.1 gives detailed information about each recording: the number of recorded neurons, the number of tone sequences presented to the animal, and the different tone frequencies used. The frequency step between two consecutive tones in the sequences was either half an octave up or down. The direction of tone change after each tone was randomly chosen, except for the maximum (minimum) frequency within a recording, where the next lower (higher) tone followed with 100% probability. The first tone on each trial was selected from a uniform distribution, which ensured that each frequency appeared approximately the same number of times within the tone sequences of one recording (except for the two extreme frequencies, which appeared about half as many times). Furthermore, each frequency appeared about equally often before the next higher and the next lower possible frequency. Tone sequences have been used previously in several studies of A1, although most focused either on two frequencies (Ulanovsky et al., 2003, 2004) or one varying frequency paired with a fixed base frequency (Brosch et al., 1999; Brosch and Schreiner, 2000). These studies reported response characteristics that were similar to those we found (see Figure 4.1).

4.2.3 Estimation of mutual information

Information contained in the neural response can be analyzed by estimating directly the *mutual information* between stimulus and response as a general measure from information theory (Shannon, 1948; Cover and Thomas, 1991). The mutual information between stimuli S (in our case, tone frequencies or direction of tone changes) and evoked responses R is given by

$$I(S;R) = H(R) - H(R|S) = \sum_{r,s} P(s)P(r|s)\log\frac{P(r|s)}{P(r)}. \tag{4.1}$$

animal	recording	# neurons	# seq.	start freq.	# freq.
Ferret 1	ele093b05	4	54	1000	9
Ferret 2	per001c06	3	35	2125	7
	per002d03	6	35	1775	7
	per003b08	3	35	1775	7
	per006a08	8	35	2510	7
	per007b10	4	35	1202	7
	per011b05	2	55	2943	11
	per011c05	1	55	2943	11
	per018c10	7	35	2050	7
	per019a05	9	35	2298	7
	per026a09	6	42	1727	7
	per027a07	10	42	354	7
	per027b06	10	42	354	7
	per028a09	6	42	1945	7
	per031a08	6	42	1768	7
	per031a09	6	54	1250	9
	per034a09	7	54	1250	9
Ferret 3	sas009b13	5	54	800	9
	sas028b07	4	54	700	9
	sas029a07	2	54	250	9
	sas031a07	6	54	625	9
Ferret 4	sag002e19	4	54	350	9
	sag005a03	3	54	725	9

Table 4.1: **Information about the 23 recordings made from the 4 ferrets.** The third column ("# neurons") contains the number of simultaneously recorded neurons. The fourth column ("# seq.") denotes the number of tone sequences (each with 100 tones) used as stimuli. For each recording the sequences consist of a series of "# freq." increasing frequency values starting with "start freq." The quotient between two neighboring frequency values was $\sqrt{2} \approx 1.41$.

S and R are random variables characterized by probability distributions $P(s)$ and $P(r)$, respectively. The conditional entropy (or noise entropy) $H(R|S)$ describes the variability of the responses for a fixed stimulus s (expressed by the conditional distribution $P(r|s)$). If this variability is small compared to the overall variability of responses $H(R)$, the response conveys a large amount of information about the stimulus. Mutual information thus quantifies the reduction of uncertainty about the stimulus that can be gained from observation of a single response trial (Rieke et al., 1997).

Calculation of mutual information requires accurate estimation of the response probabilities $P(r)$ and $P(r|s)$ from the finite experimental data, which suffers from a sampling problem: estimating the probability distribution from a finite number of observations leads to a systematic underestimation of the entropy of this distribution (Miller, 1955) and consequently to an upward bias of mutual information (Panzeri and Treves, 1996). To overcome this bias problem, we used a shuffling-based estimator (Panzeri et al., 2007), which eliminates the bias by subtracting the noise entropy of a randomly shuffled dataset. Furthermore, an additional bias

correction technique was applied that uses quadratic extrapolation of the individual entropy measures to infinite sample sizes (Panzeri et al., 2007).

To evaluate the information contained in the responses about a particular feature of the stimulus sequences (e.g., the current tone, or the direction of previous tone change) we viewed all available instances of this feature as our set of stimuli, and the spike trains in response to all occurrences of one particular instance within the tone sequences of one recording as our set of responses for this stimulus. For example, to calculate the mutual information $I(T;Y)$ between the response Y and the current tone T, the set of stimuli consisted of all available tone frequencies used in one recording, and each of these stimuli was associated with a set of responses that is composed of the neural responses during all occurrences of the respective frequency.

In order to measure the information contained in the responses about the previously played tone, we took into account the special temporal structure of the stimulus sequences. Since each tone is followed by either the tone with the next higher or the next lower frequency, there is already substantial mutual information between two consecutive tones. So a high value of the mutual information between the response and the previously played tone might actually reflect information about the current tone because it is not independently chosen from the previous tone. We therefore calculated the mutual information about the *direction* of the tone step given by the previous and current tone.

The total information that is contained in the response Y about the tone pair (T_1, T_2) can be written as, according to the chain rule of information theory (Cover and Thomas, 1991),

$$I(\Delta, T_2; Y) = I(T_2; Y) + I(\Delta; Y|T_2), \tag{4.2}$$

where Δ is a binary variable indicating whether the tone pair (T_1, T_2) is an up- or down-step in frequency. Note that the tone pair (T_1, T_2) is completely determined by the tuple (Δ, T_2). The first term, $I(T_2; Y)$, is then simply the information about the current tone (during T_2), and the second term, $I(\Delta; Y|T_2)$, measures for a given value of the current tone T_2 the *additional* information that is conveyed about the direction of the tone change from T_1 to T_2.

This approach can be extended to investigate whether the neural response contains also information about tone changes farther back in the sequence. Note that a sequence of n successive tones $(T_1, \ldots, T_n)$ is completely determined by the value of the last tone, T_n, and a sequence of $n-1$ binary variables $(\Delta_1, \ldots, \Delta_{n-1})$ indicating the directions (up or down) of the $n-1$ tone steps of the sequence $(T_1, \ldots, T_n)$. We can write analogously to (4.2)

$$\begin{aligned} I(\Delta_1, \ldots, \Delta_{n-1}, T_n; Y) &= I(T_n; Y) + I(\Delta_{n-1}; Y|T_n) + I(\Delta_{n-2}; Y|T_n, \Delta_{n-1}) \\ &\quad + \ldots + I(\Delta_1; Y|T_n, \Delta_{n-1}, \Delta_{n-2}, \ldots, \Delta_2) \\ &= I(T_n; Y) + \sum_{i=1}^{n-1} I(\Delta_i; Y|T_n, \Delta_{n-1}, \ldots, \Delta_{i+1}). \end{aligned} \tag{4.3}$$

That is, the information can be written as a sum of contributions of individual tone

changes, conditioned on later tones. Note that for $n = 2$ equation (4.2) follows as a special case.

Reliable information estimates require a suitable discretization of the response space. We discretized each spike train of a neuron as a binary word with bin size 5ms, i.e., a value of 1 in one bin indicates the presence of at least one spike in the corresponding 5ms period. At time t following stimulus onset we calculated the mutual information between the stimulus and the response of a single neuron during a time window of 20ms (4 time bins) preceding t.

We also used a different discretization scheme when we compare mutual information with classification performance. There, we used the spike count, limited to values from 0 to 4, in a time window of 20ms preceding a particular time point t to represent the response of a neuron at time t. In contrast to the binary code described before, where we calculated the information conveyed by individual neurons, we used the vector composed of the spike counts of all simultaneously recorded neurons as our response. The same vector can be used as a classifier input, which makes the comparison between mutual information and classifier performance possible.

With this discretization of the response into vectors of spike counts the size of the response space becomes potentially 5^n, where n is the number of simultaneously recorded neurons. This increases the risk of bad information estimates due to undersampling of the response space. To determine the reliability of the measured information values we evaluated the mutual information on a random subset of one half of all available trials. If the mutual information value estimated on this truncated dataset changed by less than 10% of its original value, a reliable estimate was reported.

As software we used the Python implementation described in (Ince et al., 2009).

4.2.4 Analysis of information using linear classifiers

Information in the neural response about the stimulus can also be extracted by classifiers, with spike trains of simultaneously recorded neurons as input patterns and the stimulus identity as the target label (see e.g., (Nikolic et al., 2009) for an application of this method). Here, we used (binary) linear classifiers, more precisely, Support Vector Machines (SVMs) with a linear kernel (see e.g., (Schölkopf and Smola, 2002; Bishop, 2006; Ben-Hur et al., 2008)), to predict one particular bit of information. The performance of a linear classifier can be viewed as a lower bound on the information contained in the responses about this particular bit.

All response spike trains were low-pass filtered by an exponential filter with time constant $\tau = 20$ms. At particular points in time, the values of these analog traces of multiple simultaneously recorded neurons of one particular recording form multi-dimensional input vectors that served as input patterns to the classifier. Performance of the classifiers was always evaluated with 10-fold cross validation. The parameter C of the linear SVM, which determines the trade-off between minimizing training errors and generalization, was chosen to be 100, however, we found the achieved performance not to be very sensitive to this parameter. Furthermore, we tried SVMs with different kernels (polynomial kernels of different degrees, and RBF kernels with different widths), but no significant performance increase was detected.

This means that in this case linear classifiers performed as well as most non-linear classifiers, and no significant additional information can be extracted using these non-linear classifiers.

In the XOR experiment (Figure 4.6) we measured the performance as the point-biserial correlation coefficient between the binary target variable and the continuous linear combination of the low-pass filtered input spike trains learned by the classifier (i.e., the output of the classifier *before* the threshold operation). This is necessary in order to ensure that any significant classification performance on this non-linear XOR-combination can really be attributed to a nonlinearity implicitly provided by the neural responses. It can be shown that any such correlation coefficient significantly greater than zero indicates non-linear transformations in the neural processes itself (see (Nikolic et al., 2009) for a formal proof).

When we compared mutual information with classification performance, we did not use the low-pass filtered spike trains of all simultaneously recorded neurons as input to the classifier, rather we used the vector composed of the spike counts of these neurons during the last 20ms (the same value as the time constant τ). This discretization allowed the application of both a mutual information estimator and a classifier on the same data. To convert the classification performance (% correct) into an information value (bit) we calculated directly the mutual information (4.1) between the stimulus and the classifier prediction.

As a software for solving the SVM optimization problem we used the LIB-SVM package (Chang and Lin, 2001) included in the PyML toolbox for Python (http://pyml.sourceforge.net/).

4.2.5 Data pre-processing and statistical analysis

To calculate the peri-stimulus time histograms (PSTH) for a neuron in response to a given frequency, we collected the spike times during all occurrences of this frequency during 150ms tone periods. The bin size of the PSTHs was chosen to be 1ms; for display purposes only, the PSTHs were smoothed with a 10ms-Hamming window. In order to measure correlations between two variables, we used the standard Pearson correlation coefficient, which yields an r and a p value. The r-values denote the correlation coefficient itself and p-values are the probability that these correlations are produced by random chance. Thus, a low p-value indicates a significant correlation. If r-values are reported without a p-value, $p < 0.0005$.

To assess the significance of the obtained mutual information and classification performance values we performed a label-shuffling test: We evaluated the mutual information on 100 different shuffled datasets that are generated by randomizing the stimulus identity (label) for each response. A significant information value ($p < 0.05$) was reported if it was larger than the mean plus two times the standard deviation of this distribution of shuffled information values. For the information about previously played tones we compared the average information value across different tones (or tone sequences, see equations (4.2) and (4.3)) with the distribution of the corresponding averages of the shuffled information values.

4.3 Results

4.3.1 Neural responses to tone sequences in primary auditory cortex

In order to measure the temporal integration of sensory information by A1 neurons we recorded the activity of 122 neurons isolated from 23 multi-channel recordings in four passively listening awake ferrets. The stimuli were sequences of tones of 150ms duration. The frequency step between two consecutive tones was always randomly half an octave up or down and the sequences were designed to present tones at each frequency approximately the same number of times.

Figure 4.1A shows a snapshot of such a stimulus sequence used in one recording, overlaid with the associated responses of four simultaneously recorded neurons. The peri-stimulus time histogram (PSTH) response for a given frequency was computed by averaging the spike trains following the onset of all tones at that frequency (see examples in Figure 4.1B). It can be seen that individual neurons responded to different tone frequencies in various ways. For instance, neuron #4 had a very sparse response compared to the other units. Other neurons, such as neuron #3, typically responded with a strong transient burst to a tone onset or change, whereas the responses of still other neurons, such as that of neuron #2, were sustained across the tone duration. Furthermore, especially neurons with a strong transient response tended to be more sharply tuned to specific frequencies or frequency ranges (see also Figure 4.2B).

As had been shown previously (Brosch and Schreiner, 2000; Ulanovsky et al., 2004), the responses of individual neurons to individual frequencies differed, depending on the tone frequency that had been played immediately before. Figure 4.2A shows PSTHs of four sample neurons plotted as in Figure 4.1B, but conditioned on the direction of the preceding tone step. For some neurons and frequencies, there was a substantial difference in the firing rate in response to the same stimulus frequency. This difference was particularly large for neurons responding with a strong transient burst. These context-dependent responses give rise to conditional tuning curves, which plot mean firing rate as a function of tone frequency separately for both up and down steps in frequency from the preceding to the current tone (Figure 4.2B).

This investigation of the neural responses already provides qualitative insight into the temporal integration capabilities of A1 neurons. It suggests that not only information about the currently played tone, but also about whether the previous tone was higher or lower, might be encoded in differences in both the mean firing rates across the whole tone duration and the timing of the spikes relative to the tone onset.

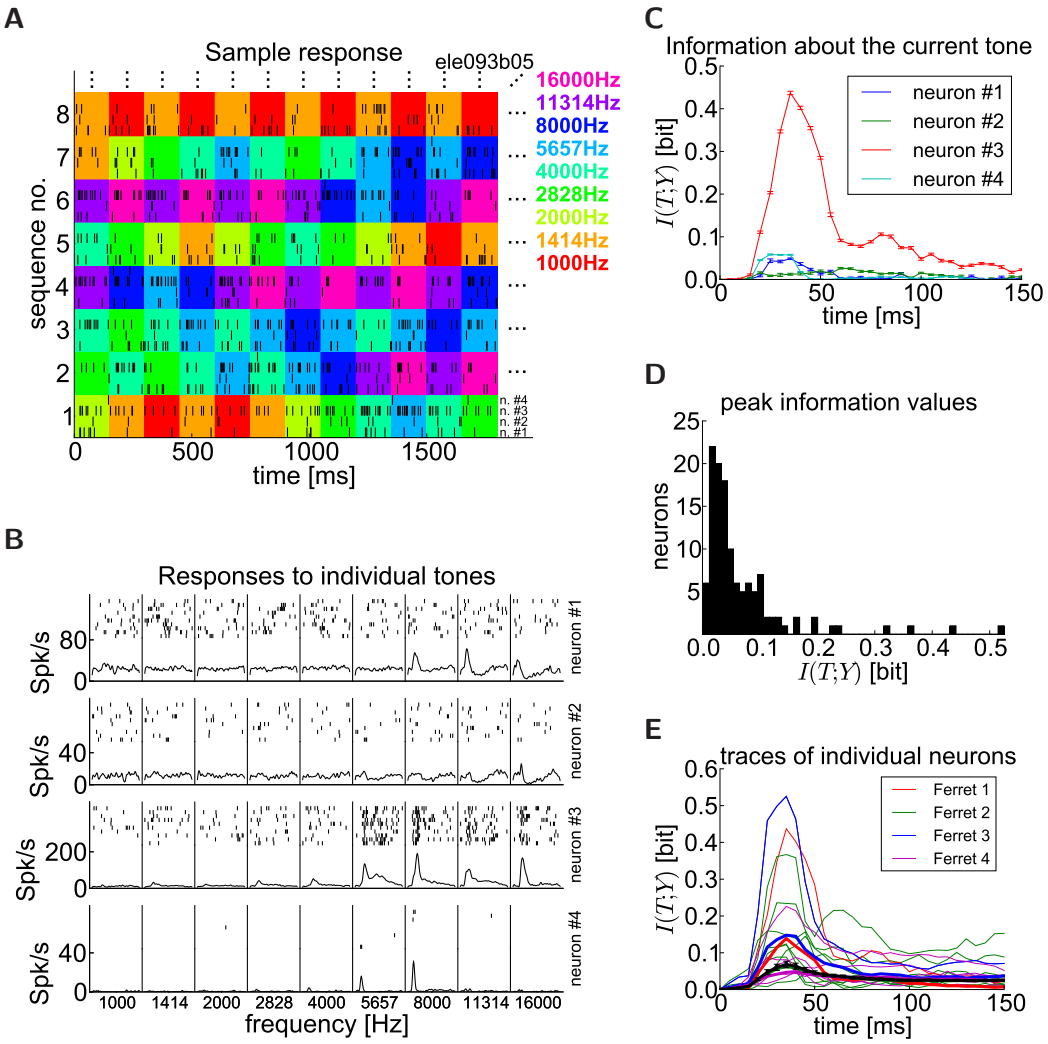

Figure 4.1: (see next page for Figure caption)

4.3.2 Direct estimation of mutual information

4.3.2.1 Most neurons convey a significant amount of information about the currently played tone

We first investigated how much information is contained in the neural responses about the tone frequency which is currently played. To address this question we measured the information $I(T;Y)$ conveyed by the responses of individual neurons Y about the frequency of the current tone T. For all available frequencies used in one recording we viewed the response spike trains of one neuron during all occurrences of that frequency as individual trials in response to the same stimulus (see Figure 4.1B). From these stimulus-response associations we then calculated the mutual information every 5ms throughout the tone duration of 150ms.

Figure 4.1: **Most neurons convey a significant amount of information about the current tone.** (**A**) Sample stimulus sequences and neural responses for the first 12 tones of the first 8 sequences played during one particular recording (Ferret 1). In total, the stimuli for this recording consisted of 54 sequences of 100 tones each. The background color indicates the frequency of the current tone, as denoted by the legend on the right. In this experiment 4 neurons were simultaneously recorded. Black lines show the spike times of the 4 different neurons (from bottom to top within one row, neurons #1 to #4). (**B**) Frequency tuning of neurons #1 to #4 in **A**. The top of each row shows 10 individual spike trains of one neuron in response to each frequency during tone presentation periods (150ms) of 10 randomly chosen occurrences of that frequency within the stimulus sequences. The bottom of each row shows the PSTHs for that neuron. Note the different scalings on the y-axis. (**C**) Time courses of mutual information between the responses of these four neurons and the currently played tone for the recording shown in **A** and **B**. MI values are estimated every 5ms from the spike train in the 20ms time window preceding the time indicated on the x-axis. Error bars denote the standard error of the mean of the MI estimator. (**D**) Histogram of peak mutual information values across the tone duration (150ms) of all 122 neurons. (**E**) Mutual information traces (as in **C**) for 16 sample neurons across all four animals, including the neurons with maximal information for each animal. The other 12 neurons were randomly selected. Thick lines: average information trace for each animal, black line: average across all neurons from all animals, error bars: standard error of the mean.

Figure 4.1C shows the time courses of information conveyed by the individual neurons shown in Figures 4.1A and 4.1B. In this recording, neuron #3 conveyed the largest amount of information about the current tone frequency. As seen in the responses to individual frequencies in Figure 4.1B this neuron was the most selective, as it responded strongly and reliably to higher frequencies. The transient response after tone onset is reflected in the time course of the mutual information, in that it reaches a relatively high peak at about 40ms after onset, decreases afterwards, but remains significant during the second half of the tone (significance assessed by a label shuffling test ($p < 0.05$), see Materials and Methods). Neurons #1 and #4 of this particular recording conveyed much less information, but the amount was still significant between 20ms and 40ms, which can be explained by their transient responses to some frequencies. Neuron #2 on the other hand, which had a rather unselective response for all frequencies, did not convey a significant amount of information, even though it responds with a higher firing rate than neuron #4. The information transmitted by other neurons from different recordings across different animals varied in time course and magnitude (see Figure 4.1E). These examples show that different neurons conveyed information in various ways.

Figure 4.1D summarizes the peak information value across the entire set of A1 neurons in our study. From the 122 neurons recorded, the responses of 94 neurons conveyed significant information about the current tone (at least at one time point during the tone interval of 150ms, a significant mutual information value was measured). The most informative neuron conveyed a peak information value of 0.53bit (measured in Ferret 3), but this value lay below 0.1bit for most of the

neurons. The average peak information across all neurons was 0.067bit and was not significantly different between animals (Ferret 1: 0.143bit, Ferret 2: 0.060bit, Ferret 3: 0.124bit, Ferret 4: 0.066bit). Although the average information conveyed by single neurons was substantially less than the theoretical maximum (2.75bit, 3.12bit, or 3.42bit, depending on whether 7, 9, or 11 frequency values were used in the recording), this information can accumulate over large numbers of neurons in A1 to accurately encode the stimulus.

Neurons that conveyed large amounts of information also tended to have high firing rate responses. The information conveyed by individual neurons and their average firing rate were correlated. The value of Pearson's correlation coefficient between mutual information and mean firing rate was significantly positive ($r = 0.69$) for the recording shown in Figures 4.1A-C. The overall correlation coefficient for all responses was $r = 0.50$.

4.3.2.2 Neurons simultaneously convey information about the current and previous tone

In order to look for evidence of stimulus integration over time, we next measured the information contained in the responses of individual neurons about the previously played tone. A high mutual information between the current response and the previous tone indicates a strong temporal integration capability of this neuron. We measured the information value $I(\Delta; Y|T_2)$ between the response Y and the direction of the tone change (up or down, indicated by the binary variable Δ) preceding the current tone T_2 (equation (4.2) in Materials and Methods). Because each tone was followed by either the next higher or the next lower tone, all information about the previous tone, given the current tone, was captured by this single-bit value.

Figure 4.2 illustrates the calculation of information between the response and the direction of the previous tone change Δ in detail for four neurons. The responses of these neurons differed substantially in their response to some frequencies, depending on whether the higher or lower tone was played immediately before (Figure 4.2A). This difference was particularly large in the transient responses. The conditional tuning curves in Figure 4.2B show that these differences in the mean firing rates for a certain neuron were most prominent for the range of preferred frequencies of that neuron. Figure 4.2C shows the time course of information about the frequency change, $I(\Delta; Y|T_2)$, for each of these 4 sample neurons for different values of T_2. It can be seen that neurons encoded this information in various ways. The most information was contained during the transient response, but it could sometimes persist for the duration of the tone. The amount of information conveyed also varied with the current tone T_2.

Figure 4.3A compares the average peak information about current and previous tones across the entire set of A1 neurons. From all the 122 neurons recorded, the responses of 41 neurons conveyed significant information about the direction of the previous tone change Δ (at least at one time point during the tone interval of 150ms, the average information across tones T_2 was significant). On average, the ratio between the information values was 74:26, i.e., the information value $I(T_2; Y)$ was about three times larger than $I(\Delta; Y|T_2)$. The maximal peak information value

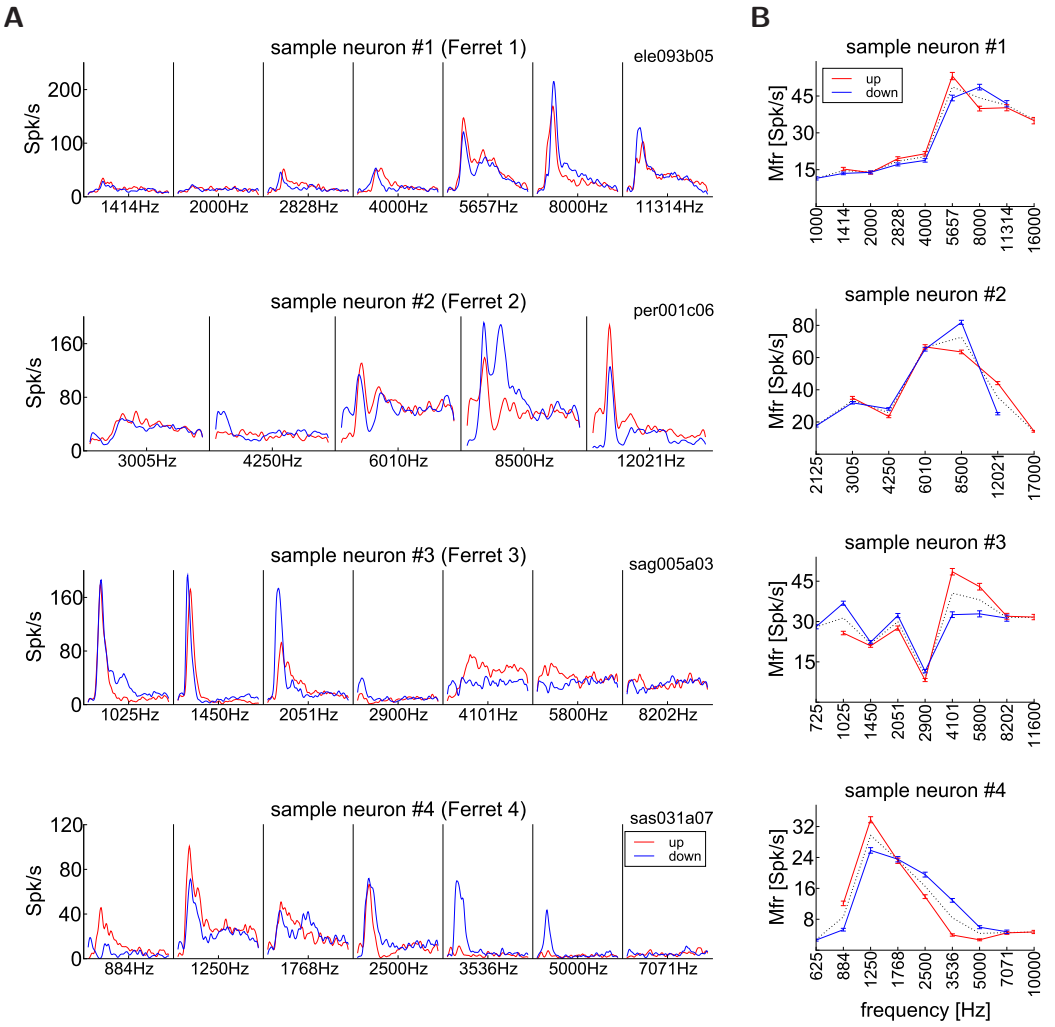

Figure 4.2: (continues on next page)

measured about Δ was 0.158bit in Ferret 1, 0.391bit in Ferret 2, 0.378bit in Ferret 3, and 0.306bit in Ferret 4. The average peak information was 0.045bit (Ferret 1: 0.045bit, Ferret 2: 0.040bit, Ferret 3: 0.062bit, Ferret 4: 0.058bit). These values were not significantly different between animals. Note that the maximum possible value for the information about Δ is 1bit, whereas the maximum information about the current tone, as measured in Figure 4.1, is the entropy of the distribution of tone frequencies in the respective recording (2.75bit, 3.12bit, or 3.42bit, depending on whether 7, 9, or 11 frequency values were used in the recording).

Most of the information about the change from the previous tone can be explained by the firing rate differences in response to the two possible predecessors of a given tone (see examples in Figure 4.2A and 4.2B). There was a significant corre-

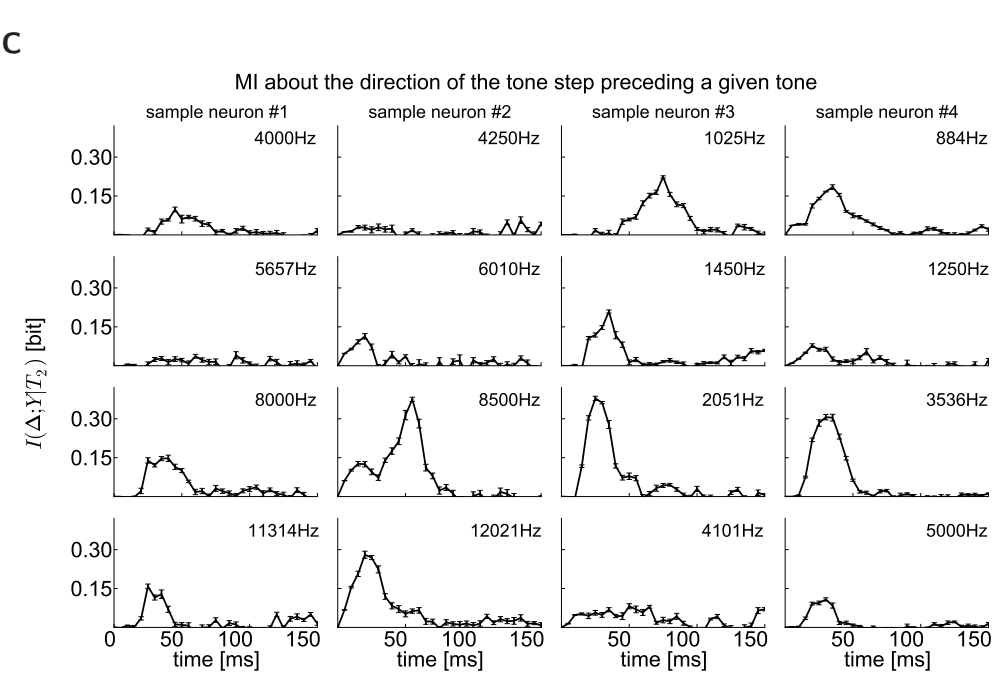

Figure 4.2: **Many neurons convey significant information about the direction of the preceding tone step.** (**A**) The PSTHs of 4 sample neurons (one for each Ferret) to each tone frequency are substantially different, depending on whether the frequency of the previously played tone was higher (down step, blue) or lower (up-step, red). (**B**) Conditional tuning curves for the sample neurons in **A** show this difference in the mean firing rate of these neurons during 150ms periods in response to all available frequencies, depending on whether the next lower or the next higher frequency has been played immediately before. The black dotted line shows the overall tuning curve calculated from all responses independent of the previous tone. Error bars denote SEM. (**C**) Time course of mutual information $I(\Delta;Y|T_2)$ about the direction of the tone change (Δ) preceding a given frequency during the current tone, T_2. Columns of panels correspond to different recordings, rows correspond to different values of the current tone. MI values are estimated every 5ms from the spike train in the 20ms time window preceding the time indicated on the x-axis. Error bars denote the standard error of the mean of the MI estimator.

lation between the peak value of the mutual information $I(\Delta;Y|T_2)$, and differences in the mean firing rate across the whole tone duration (Figure 4.2B). The overall correlation coefficient was $r = 0.729$, and this correlation was also significant for individual animals (Ferret 1: $r = 0.483$, $p = 0.009$; Ferret 2: $r = 0.798$; Ferret 3: $r = 0.491$; Ferret 4: $r = 0.768$; all $p < 0.0005$).

Information values were often high for frequencies where the difference in response to the previous tone was large (e.g., sample neuron #2, 8500Hz), but also in cases where this difference was not particularly large (e.g., sample neuron #3, 2051Hz). Vice versa, a large response difference does not necessarily imply a larger amount of information (e.g., sample neuron #4, 1250Hz, or sample neuron #3, 4101Hz). This indicates that some, but not all, of the available information about

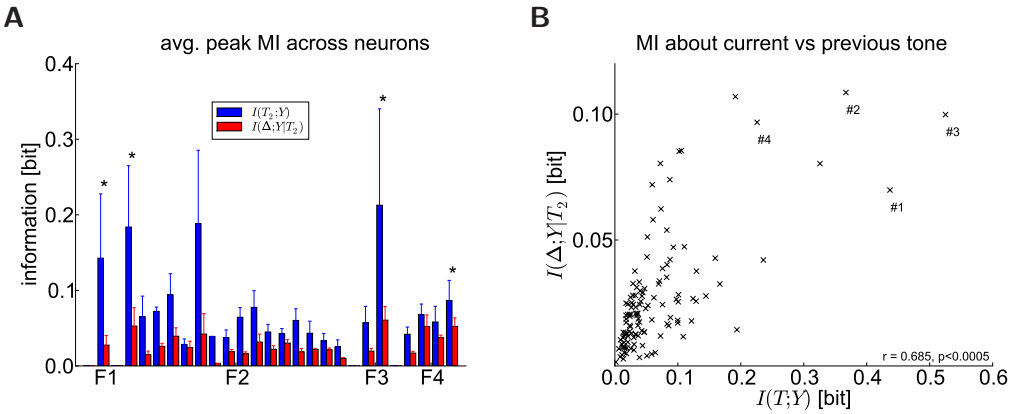

Figure 4.3: **Neurons simultaneously convey information about the current and previous tone.** (**A**) Comparison of peak mutual information values measured about the current tone (as in Fig. 4.1C, blue) and about the direction of the previous tone change (as in Fig. 4.2C, red). For each of the 23 recordings, the average across neurons (and tones) are shown. On average, the ratio of these information values is about 3:1. Error bars denote SEM; recordings are grouped by animals (F1 to F4: Ferret 1 to 4). The recordings marked with an asterisk contain the sample neurons shown in Fig. 4.1. (**B**) Neurons simultaneously transmit information about current and previous tone. The scatter plot compares the peak mutual information values about current and previous tones for each of the 122 neurons. The sample neurons in in Fig. 4.1 are marked with labels "#1" to "#4".

the previous tone is encoded in the mean firing rates of neurons. Also the timing of the spikes relative to the tone onset is important. For example, sample neuron #3 conveyed a large amount of information about the tone preceding 2051Hz because it responded with a stronger transient to a down-step (Figure 4.2A), even though the mean firing rate across the whole tone duration was similar in both cases.

Figure 4.3B shows that the peak value of the information conveyed about the current tone, $I(T_2;Y)$, and the peak value of the information about Δ given the current tone, $I(\Delta;Y|T_2)$, are significantly correlated. This means that neurons, which transmitted a large amount of information about the current tone, simultaneously also tended to convey a considerable amount of information about the direction of the tone change that had led to this current tone. This indicates that there are no neurons which "specialize" on either current or previous tones, but rather that individual neurons really integrate information about previously arrived stimuli into their current responses.

4.3.2.3 Temporal integration of information about earlier tones

To investigate whether neurons integrate information also about tone changes farther back in the sequence, we extended the approach in the previous section. We used the chain rule of information theory to evaluate the information about tones an increasing number of time steps in the past. More precisely, we calculated infor-

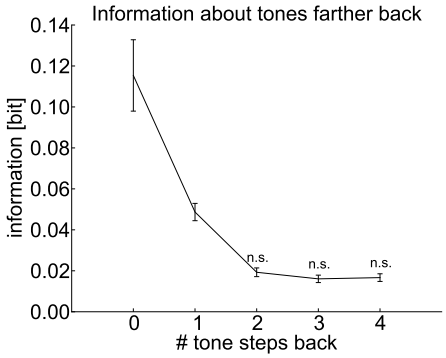

Figure 4.4: **No additional information can be gained about the direction of the tone step more than two tones back.** Mutual information values are calculated between the response and the direction of the tone change the specified number of time steps in the past. The average across neurons and sequences (see text) of the peak information during a 150ms tone duration is shown. The labels on the abscissa denote the number of time steps back (0: information about the current tone, 1: information about the previous tone, 2: two steps back, etc). Error bars denote SEM; n.s., non-significant deviation from chance level, estimated by a label-shuffling test ($p > 0.05$).

mation values $I(\Delta_i; Y|T_n, \Delta_{n-1}, \ldots, \Delta_{i+1})$ between the response Y and the binary variable Δ_i specifying whether the direction of the tone step i steps before the current tone T_n was up or down, conditioned on all the subsequent tones up to the current tone T_n (see equation (4.3) in Materials and Methods). Note that for $n = 2$ and $i = 1$ this term is equal to the information about the direction of the preceding tone change, $I(\Delta; Y|T_2)$, considered in the previous section.

In Figure 4.4 we compared the average of these information values for 1-4 tone steps back across neurons and across sequences $T_n, \Delta_{n-1}, \ldots, \Delta_{i+1}$. We included only those neurons which conveyed a significant amount of information about the direction of the previous tone change (n=41/122). For these 41 neurons, the average information values were higher than the average across all 122 neurons: The average information about the current tone was 0.115bit and the average information about the direction of the tone change preceding the current tone was 0.049bit. The responses of 25, 12, and 6 neurons conveyed significant information about the tone change Δ_2, Δ_3, and Δ_4, respectively (at least at one time point during the tone interval of 150ms, the average information across sequences $T_n, \Delta_{n-1}, \ldots, \Delta_{i+1}$ was significant). It can be seen that the information decreases for increasing number of tone steps considered. After two tone steps the information saturates at a non-significant low residual value. This would indicate that the response does not contain any information about the tone step direction more than two tones in the past. However, we are exploring a large stimulus space because we have to average over all possible sequences of length i. Thus, the number of available samples for mutual information estimation decreases by one half for each additional tone step considered. Therefore the information values for many backward time steps are less reliable and we cannot exactly measure the influence of earlier tones.

4.3.3 Information analysis using linear classifiers

4.3.3.1 Temporal integration of information about the previous tone

An alternative way to investigate the temporal integration of information is to measure the amount of information about past stimuli that can be extracted by a neuron that reads the current activity of all simultaneously recorded neurons. To analyze this information, we trained (binary) linear classifiers, Support Vector Machines (SVMs) with a linear kernel (see Materials and Methods), on the spike trains of simultaneously recorded neurons to decode information about current and previous tones.

First, we investigated the performance of these classifiers on the current tone. We selected two specific frequency values and collected the responses to all occurrences of these two frequencies. We low-pass filtered the spike responses ($\tau = 20$ms), and at every 10ms, we trained a different classifier to discriminate between the two possible current tones. For most tone pairs performance was significantly above baseline level of 50% during the whole duration of 150ms, across different recordings and animals. 17 of 23 recording sites performed significantly above chance (on average across all tone pairs). Peak classification performance (up to 90%) was often achieved within the first 50ms, most probably due to the discriminative initial bursting behavior of some neurons. Performance typically decreased for later time points. The average peak performance across all tone pairs and experiments was 59.75% (Ferret 1: 65.23%, Ferret 2: 58.38%, Ferret 3: 62.46%, Ferret 4: 61.21%). In principle, one could also investigate the performance of multi-class classifiers on all available tone frequencies, but since they are essentially a combination of different linear classifiers, they do not report additional information.

We then used linear classifiers to investigate the temporal integration of information, i.e., the information contained in the responses about the previously played tone. Figure 4.5A shows examples of performance of linear classifiers trained on all simultaneously recorded neurons to discriminate between the two possible predecessors of the current tone, i.e., to extract information about Δ in equation (4.2). Performance is also shown during the preceding tones. Every 10ms a different classifier was trained to discriminate between the two different frequencies of the first tone T_1 in the pair (T_1, T_2) for a given tone T_2. In this way we could analyze the information extracted by classifiers about which of two different frequencies was currently played, and how this information is maintained during the next tone. One sees that in most cases performance stays above chance level for some time after the switch of the tone frequency before it drops, indicating that the information is maintained about the stimulus after the tone has switched. In other cases, performance reached a second (albeit smaller) peak after the tone switch. This is most probably to differences in the transient responses of some neurons.

Figure 4.5B summarizes the average peak classifier performance for each dataset (averaged over tones T_2 of the maximum performance in extracting information about T_1 during the period of T_2). Classifiers were able to extract a considerable amount of information about the previous tone across different frequencies for most recordings. 18 of 23 recording sites performed significantly above chance (on average across all tones T_2). The maximal measured performance achieved was 88.17%

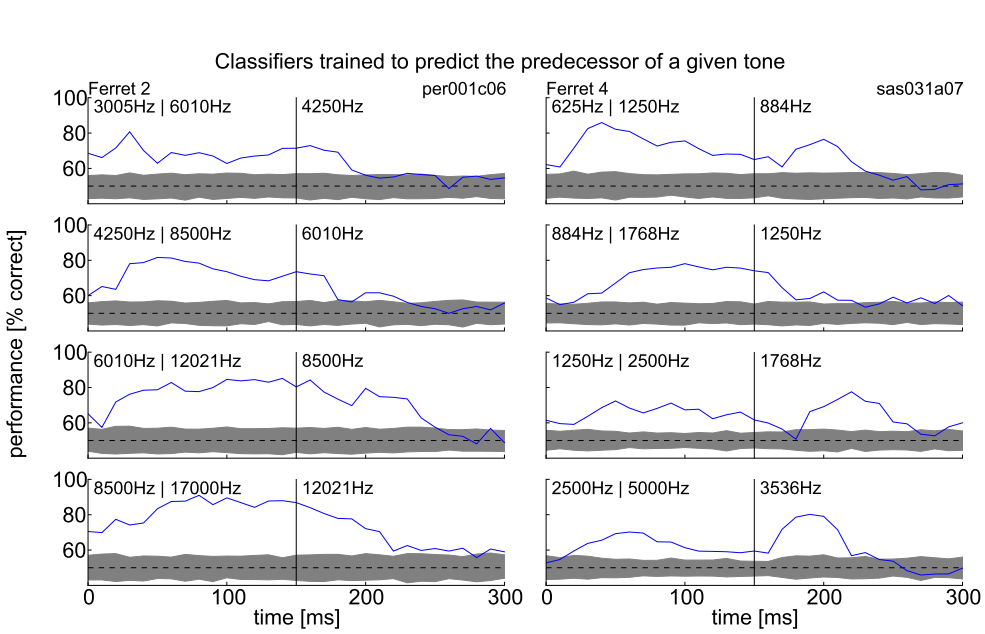

Figure 4.5: (continues on next page)

(in Ferret 2, shown in the first column of Figure 4.5A). The average peak performance was 64.92% (Ferret 1: 59.18%, Ferret 2: 64.40%, Ferret 3: 63.86%, Ferret 4: 68.92%). Note, however, that direct comparisons of the classification performance between different recordings should be viewed with caution, since the performance values depend on several factors such as the number of neurons simultaneously recorded or the total number of spikes in the recording. Similar to the mutual information values about the previous tone, peak classification performance was also strongly correlated with the absolute firing rate differences of individual neurons (Figure 4.2B). The overall correlation coefficient was $r = 0.394$ (Ferret 1: $r = 0.483$, $p = 0.192$; Ferret 2: $r = 0.411$, $p < 0.0005$; Ferret 3: $r = 0.459$, $p = 0.001$; Ferret 4: $r = 0.272$, $p = 0.003$).

These performance values of linear classifiers on the previous tone are on average similar or even slightly higher than the performance values on the current tone. This demonstrates the prominent temporal integration capability of A1 neurons. Note that this finding does not contradict the previously reported 3:1 ratio of mutual information about current versus previous tone, because the mutual information about the current tone is evaluated using all available frequencies at the same time, while for the classifier only pairs of frequencies are considered.

4.3.3.2 Non-linear superposition of information

Furthermore we analyzed whether the neural response provides a non-linear superposition of information about sequentially arriving stimuli. This property is beneficial for information processing because it boosts the computational power

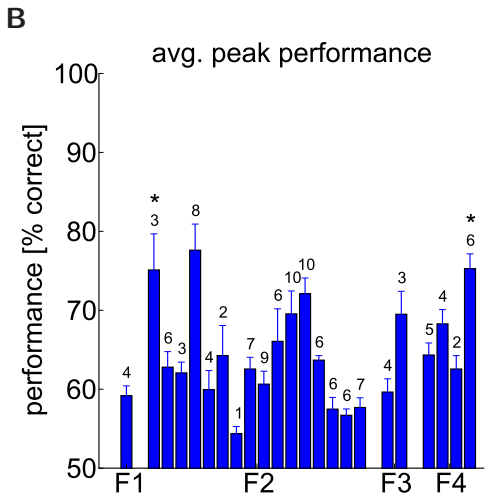

Figure 4.5: **Linear classifiers are able to discriminate between the two possible predecessors of a given tone.** (**A**) Performance of linear classifiers trained at time points every 10ms during the 300ms interval of two consecutive tones T_1, T_2 to discriminate between the two possible predecessors T_1 of a given tone T_2 (i.e., to predict Δ) for two example recordings from two different animals. Columns correspond to different recordings, rows correspond to different values of T_2. Performance is evaluated by 10-fold cross validation. The dashed black line indicates baseline performance of 50%. The gray shaded stripe around the baseline performance denotes the region of non-significant deviations from chance level, estimated by a label-shuffling test ($p > 0.05$). (**B**) Average peak performance values: For each of the 23 recordings, the peak classification performance during the duration of the second tone T_2 is shown (average across frequencies). Numbers denote the dimensionality of the classifier input (i.e., the number of simultaneously recorded neurons). Error bars denote SEM; recordings are grouped by animals (F1 to F4: Ferret 1 to 4). The recordings marked with an asterisk are shown in **A**.

of a neuron that reads this neural response (like a kernel in the terminology of machine learning). In this way a linear neuron is effectively able to compute a non-linear function of the input. Such non-linear superposition can be proved if a linear classifier is able to reproduce the exclusive-or (XOR) computation of two bits in the input sequence. This simple binary computation yields result "1" if these two bits are different and "0" if they are the same, but it cannot be solved by any linear model. If a linear decoder is able to predict the resulting bit of information from the responses, the neural system itself has to provide the necessary non-linear combination. In (Nikolic et al., 2009) such an analysis was performed for neural responses of the primary visual cortex of anesthetized cat, and it was shown that these responses exhibited a significant non-linear combination of sequentially presented stimuli in the visual field. In our case we evaluated the XOR-performance of a classifier for our particular stimulus sequences in the following way: We collected responses to subsequences ABA, ABC, CBA, and CBC of 3 tones A, B, C (note that a tone B is both preceded and followed by either tone A (lower) or C (higher)) and check whether a linear classifier is able to predict the bit that results from the

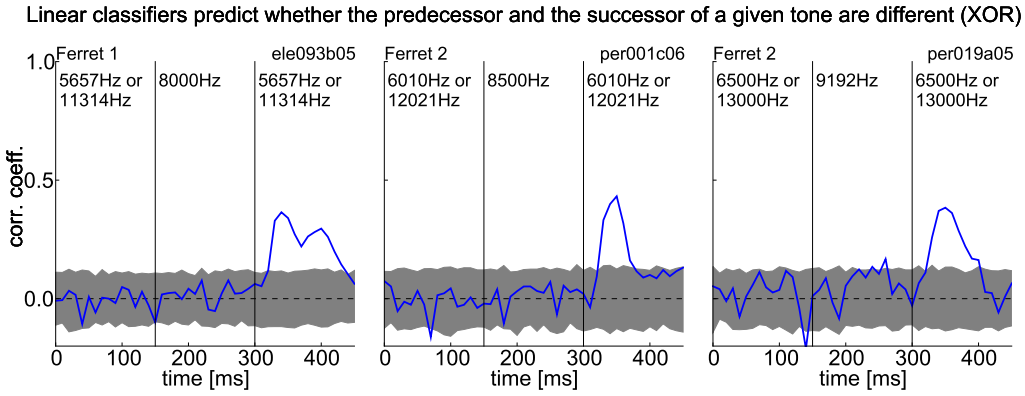

Figure 4.6: **Linear classifiers reveal a non-linear superposition of information.** Performance of linear classifiers trained every 10ms to predict a non-linear XOR-computation on the input sequence: whether the first and the third tone of a three-tone sequence was equal or not, for a fixed frequency of the intervening tone. Shown is the performance from one frequency of three different recordings from two different ferrets. Here, the performance is assessed by the point-biserial correlation coefficient between the binary target variable and the continuous readout "depolarization". The dashed black line indicates baseline correlation of 0. The gray shaded stripe around the baseline performance denotes the region of non-significant deviations from chance level, estimated by a label-shuffling test ($p > 0.05$).

XOR-combination of whether the first and the third tone in these subsequences was equal or not, for a fixed frequency of the intervening tone.

In order to ensure that any significant classification performance on this non-linear XOR-combination can really be attributed to a nonlinearity implicitly provided by the neural responses and not to the non-linear classification threshold, we measured the performance as the point-biserial correlation coefficient between the binary target variable and the continuous linear combination of the low-pass filtered input spike trains learned by the classifier (Nikolic et al., 2009) (i.e., the output of the classifier *before* the threshold operation). It can be shown that any such correlation coefficient significantly greater than zero indicates non-linear transformations in the neural processes itself (see (Nikolic et al., 2009) for a formal proof). Figure 4.6 shows that there is a significant performance increase above chance level for a considerable period during the the third tone in the sequence for at least three different recordings of two different animals, and for some values of tone B. The performance of most recordings, however, was only slightly above the significance level for only a limited number of time points, and often only for a single tone. Note that this finding provides evidence for both non-linear superposition and temporal integration of information because a past stimulus is involved in the non-linear computation.

4.3.3.3 Linear decoders are able to extract most of the total information

Finally, we addressed the question how much of the total information about the stimulus is accessible to linear decoders, and thus to hypothetical readout neurons

A

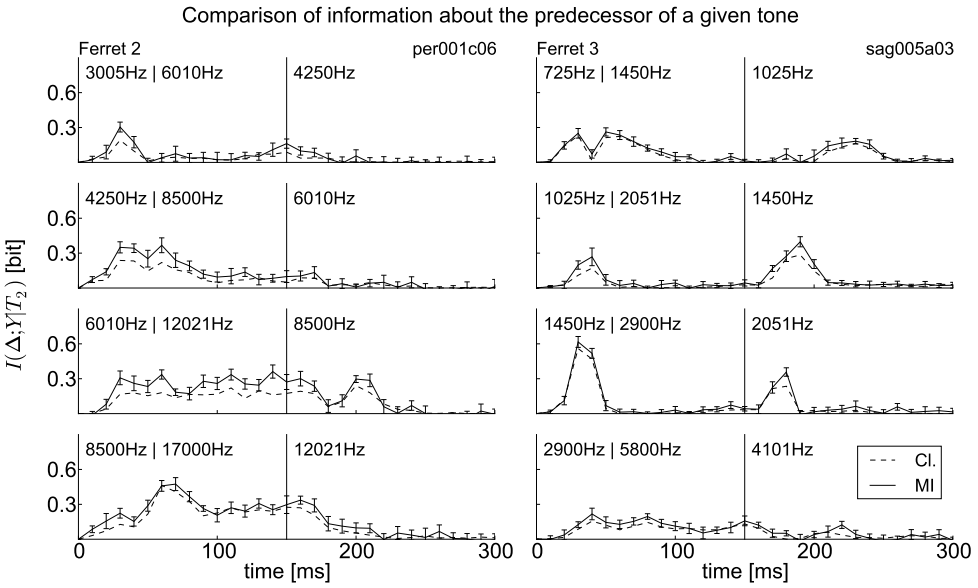

Figure 4.7: (continues on next page)

of A1 responses. For that we compared the direct estimation of mutual information with the achieved performance of a linear classifier. For a direct comparison we used the vector composed of these spike counts (20ms sliding window) for all simultaneously recorded neurons as input to both the linear classifier and for mutual information estimation. Note that, in contrast to the previous mutual information measurements, we calculated here the *combined* information conveyed by all simultaneously recorded neurons, instead of individual neurons. The mutual information between the classifier prediction and the actual stimulus could then compared to the mutual information directly estimated from the vector of spike counts.

Figures 4.7A shows such a comparison for the information between the response and the direction of the preceding tone change (as in Figures 4.2 and 4.5). Linear classifiers are trained to predict from the spike count information in response to two successive tones T_1, T_2 which of the two possible predecessors T_1 of a given tone T_2 has been played as the first tone. This information is compared to the direct mutual information estimation. It can be seen that this amount of total information is higher than the information of the classifiers throughout the whole duration of the tone pair, which is obvious since a linear classifier can extract only a subset of the total amount of information. What is interesting is that most of this available information is accessible to a linear classifier. This also holds for the information about the tone currently played, where the information is evaluated between the response and which of two selected frequencies are currently played.

Figure 4.7B shows that in almost all recordings across different animals the mutual information analysis and the training of linear classifiers extract roughly the same amount of information, although the direct estimation of mutual information

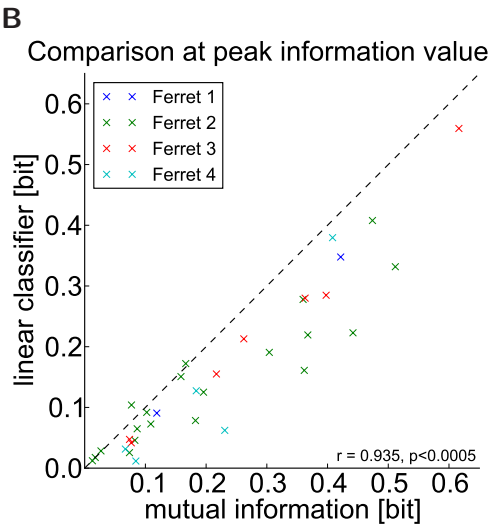

Figure 4.7: **Most of the total information can be extracted by a linear classifier.** (**A**) Time courses of information estimated directly ("MI", solid) and through the performance of a linear classifier ("Cl.", dashed) conveyed by all simultaneously recorded neurons about the direction of the tone step preceding a given tone T_2. Most of the time the classifier trace stays close below the mutual information trace. Information values are estimated every 10ms during the 300ms interval of two consecutive tones T_1, T_2 for recordings from two different animals using the sequence of spikes in the 20ms time window preceding the time indicated on the x-axis. Error bars denote the standard error of the mean of the MI estimator. Columns of panels correspond to different recordings, rows correspond to different values of T_2. (**B**) Each point in the scatter plot compares mutual information calculated directly from the neural response and from the output of the linear classifier for experiments, where mutual information could be reliably estimated. Information values are taken at the time point of maximum information throughout the 300ms duration shown in **A**. Most points lie slightly under the diagonal (dashed) indicating that much of the information is accessible to linear classifiers (median ratio 0.689, Pearson correlation $r = 0.935$).

also captures information that cannot be extracted by a classifier. The median ratio between both information values was 0.689, i.e., about 69% of the information is accessible to linear classifiers. The correlation between both information values was $r = 0.935$, i.e., for neural responses that contained much information linear classifiers also tended to decode a lot of information, and this effect was consistent across recordings. However, several recordings did not allow a direct comparison because no reliable estimate for the combined information of all neurons could be generated. This is because the response space is undersampled, either because there were too few occurrences of a given frequency (pair) or the response space was too large (too many simultaneously recorded neurons). Figure 4.7B shows only mutual information values for such recordings and frequencies where a reliable mutual information estimation was possible. The reliability of an estimate was determined by evaluating the mutual information on a random subset of one half of all available trials. If the mutual information value changed by less than 10% of

its value, a reliable estimate was reported.

This suggests that A1 provides an effective generic preprocessing of the temporal structure of acoustic stimuli and provides a suitable neural response that makes this information available to higher areas that can easily read out this information, e.g., via a simple linear neuron.

4.4 Discussion

In this work we investigated the ability of the auditory system to integrate the rich temporal structure of the acoustic environment into a neural response which facilitates further processing of the stimulus information by higher areas. We quantified the temporal integration capability of neurons in the primary auditory cortex of awake ferrets by measuring the information contained in the neural responses about both past and present stimuli.

4.4.1 A1 neurons integrate information about current and preceding tones

We compared two methods for quantifying the impact of the immediate history of auditory stimulation onto the neural responses in primary auditory cortex: the direct estimation of mutual information between the response and stimulus parameters (Panzeri et al., 2007) and the training of linear discriminators to decode stimulus parameters (Nikolic et al., 2009). To the best of our knowledge, this is the first study that directly compares the analysis of multichannel spike data via mutual information with an analysis using linear classifiers. Our stimuli, which consisted of tone sequences changing by a fixed step up or down, captured dynamic features typical of many naturally occurring sounds for which both the current and preceding frequency are important for accurate perception (Singh and Theunissen, 2003). The fact that we observe information about tone history encoded in the neural responses implies that neurons in A1 would also be able to integrate information about the more complex spectro-temporal dynamics that occur in natural sounds, thus producing a representation that would enable discrimination of different sounds. Our use of a wide range of stimulus frequencies allowed us to use information metrics effectively and to build on previous studies of auditory context (McKenna et al., 1989; Margoliash and Fortune, 1992; Lewicki and Arthur, 1996; Doupe, 1997; Kilgard and Merzenich, 1999; Brosch et al., 1999; Brosch and Schreiner, 2000; Malone et al., 2002; Ulanovsky et al., 2004; Bartlett and Wang, 2005; Yin et al., 2008; Asari and Zador, 2009). In spite of the large variability in the average responses of different neurons, both methods revealed that many neurons simultaneously conveyed information about both the current sound (77%) and the immediately preceding sound (34%).

While most neurons transmitted a significant amount of information, the absolute value of the maximal information varied substantially and tended to be lower in neurons with very sparse responses (DeWeese et al., 2003, 2005; Hromádka et al., 2008). Moreover, we found that the same neurons that conveyed a large amount of

information about the current tone also conveyed a considerable amount of information about the previous tone. This supports the hypothesis that auditory neurons integrate information about preceding sounds into their current responses, rather than encoding these pieces of information separately between different populations.

A large amount of the measured information about both the current and the previous tone was encoded in the mean firing rates of neurons rather than in their fine spike timing. Neurons that showed a clear tuning to frequency typically conveyed a larger amount of information. However, coarse timing of neural responses was also crucial. As in many previous studies, we observed both transient and sustained responses (Lu et al., 2001; Wang et al., 2005; Hromadka and Zador, 2009). Some neurons showed a strong transient response to a tone change that is beneficial for information transmission. Moreover, responses that differed in the strength of their transients for different stimuli often contained a large amount of information, even if the mean firing rate across the whole tone duration was similar for these stimuli. Recent evidence suggests that the role of precise spike timing is more prominent for stimuli varying on a faster time scale and therefore depends on the particular stimulus dynamics (Kayser et al., 2010).

4.4.2 The primary auditory cortex provides a generic preprocessing for higher areas

Our comparison of information and classifier methods revealed that the information contained in the responses about current and preceding tones is largely accessible to linear classifiers. This suggests that the primary auditory cortex provides a neural response that facilitates the task of later processing stages to instantaneously read out information about the complex temporal structure of the acoustic stimuli. Linear decoders showed a significant performance above chance level when trained to discriminate between the two possible predecessors of a given tone. The amount of information that could be decoded in a linear manner was close to the value obtained by direct estimation of mutual information, indicating that classification performance provided a rather good estimate of the total information contained in the responses. Such efficiency of linear decoding mechanisms has been previously reported, e.g., in the fly visual system (Rieke et al., 1997).

Moreover, information about current and previous tones did not occur by a simple linear superposition of information. Instead, the neural response appears to be a non-linear combination of sequentially arriving inputs, as revealed by the successful XOR classification. The XOR computation is a very simple function that tells whether the current tone and the tone two steps back are the same, for a fixed frequency of the intervening tone. However, this problem cannot be solved by a linear model. The fact that linear decoders are able to predict the resulting bit of information from the responses suggests that the neural system itself provides the necessary non-linear combination. Evidence for such non-linear interactions has recently been reported (Sadagopan and Wang, 2009), where A1 neurons were found to be sensitive to non-linear combinations of spectral and temporal stimulus properties.

However, neither method revealed that A1 neurons encode significant informa-

tion about the direction of the previous tone step, *independent* of the frequency of the current tone. Detection of frequency changes independent of base frequency may be important for some behaviors (Yin et al., 2010), but the information must be extracted from stimuli in other cortical areas.

4.4.3 Novel experimental evidence for the liquid computing model

Liquid computing (Maass et al., 2002; Buonomano and Maass, 2009) has emerged as a framework for understanding computations in biological networks of neurons, which could explain the observed response characteristics. This model proposes two fundamental operations of neural circuits: to provide (i) analog fading memory to accumulate incoming information over time in the current neural activity, and (ii) a non-linear projection into a space of typically higher dimension than the input space. With this generic preprocessing, even simple static linear neurons that "read" only the neural response at one point in time are able to extract information about the stimulus that is non-trivially spread out in time. Our analysis revealed significant evidence for both of these operations: sequentially arriving stimulus information is integrated over time and superimposed in a non-linear manner into the neural responses at one point in time. Already at this early stage of sensory processing the neural system transforms the auditory information in a way that eases the extraction of information by later processing stages. This is further supported by our finding that information extracted by linear classifiers is close to the total mutual information between stimulus and response. Besides, we did not achieve a significant performance improvement by using various types of non-linear classifiers, a further indication that most of the information contained can actually be extracted by linear classifiers.

It is essential to note that we report here results from awake animals, where the sensory machinery is operating under natural conditions. In (Nikolic et al., 2009) similar observations were made in the primary visual cortex of anesthetized cats, but the effects of anesthesia on the effects reported there are unclear. Moreover, the data from that study did not permit the direct application of standard information measures because the relevant information was spread over too many neurons. Our findings could lay the ground for a new paradigm of data analysis: For neural systems that show such generic preprocessing behavior, the computationally very efficient analysis of information using linear classifiers provides an almost as good estimate for the actual information than the tedious and biased direct estimation of information between stimulus and response.

4.4.4 Mechanisms for integration of stimulus history into neuronal responses

A number of possible mechanisms could allow information about previously played tones to persist in the responses to the current tone. First, this persistence could be implemented in an earlier auditory area responsible for some kind of echoic memory. Contextual effects have been reported in subcortical areas (Anderson et al., 2009), but in this case some mechanism must then explain the emergent

effect in those areas. Second, the A1 neurons themselves could "remember" the influence of previous inputs in their biophysical state, e.g., by mechanisms of short-term plasticity of their synapses (Zucker and Regehr, 2002; Elhilali et al., 2004; Wehr and Zador, 2005; David et al., 2009). For example, it has been shown in (Wehr and Zador, 2005) that the influence of synaptic depression on responses can last for several 100 ms. Third, the integration of stimulus history into the neuronal responses could be implemented by lateral (possibly inhibitory) inputs from other cortical neurons. Information about a preceding tone could be contained in the activity of inhibitory inputs, which then shape the subsequent responses to the current tone. There are many different possible intracortical pathways, and it could also be possible that these processes involve inputs from higher cortical areas.

Neuronal adaptation is a related mechanism that has been extensively studied in the auditory context (see e.g., (Condon and Weinberger, 1991; Malone and Semple, 2001; Malone et al., 2002; Ulanovsky et al., 2003, 2004)). One effect of neuronal adaptation is to maximize information transmission by matching the neural code to the stimulus statistics (Fairhall et al., 2001). In (Ulanovsky et al., 2004) such stimulus-specific adaptation (SSA) has been studied for stimulus sequences of pure tones. This study found evidence that A1 responses are influenced by tones up to four or five steps in the past. While we found a similar effect in our data, we had limited statistical power to measure information about stimuli more than two steps back in time because we sampled frequencies over a large range of the tuning curve of the neurons in order to estimate mutual information. Given the previous reports of SSA in A1 and other contextual effects in V1 (Nikolic et al., 2009), such long lasting dependence of information on preceding stimuli is likely.

On the other hand, the influence of earlier stimuli on the current neural responses is probably affected by anesthesia, e.g., it might change the impact of inhibitory neurons so that they are not able to reset cortical circuits after a stimulus as they might do in awake animals. In contrast to the related studies (Ulanovsky et al., 2004) and (Nikolic et al., 2009), we report here results from awake animals, and it is possible that the long lasting persistence of information reported there is at least partly an effect of anesthesia.

4.5 Acknowledgements

This Chapter is based on the paper *Neurons in primary auditory cortex integrate information about past and present stimuli* by myself (SK), Stephen V. David (SVD), Pingbo Yin (PY), Shihab A. Shamma (SAS), and Wolfgang Maass (WM). The biological experiments were designed by SVD, PY, and SAS, and conducted by SVD ad PY. The data was analyzed by SK; SVD and WM provided additional ideas for the data analysis. The paper was written by SK, with additional input from SVD and WM. Stefan Häusler and Stefano Panzeri provided stimulating comments and discussions. This work was supported by grants from the National Institute for Deafness and Other Communication Disorders (grants R01DC005779 and F32DC008453), and project # FP6-015879 (FACETS) of the European Union.

Appendix A

List of Publications

1. S. Klampfl. *Extracting statistically independent components with a generalized BCM rule for spiking neurons.* Diploma thesis, Institute for Theoretical Computer Science, Graz University of Technology, 2006.

2. S. Klampfl, R. Legenstein, and W. Maass. *Information bottleneck optimization and independent component extraction with spiking neurons.* In Proc. of NIPS 2006, Advances in Neural Information Processing Systems, volume 19, pp. 713-720. MIT Press, 2007.

3. S. Klampfl, R. Legenstein, and W. Maass. *Spiking neurons can learn to solve information bottleneck problems and to extract independent components.* Neural Computation, 21(4):911-959, 2009.

4. S. Klampfl, S. V. David, P. Yin, S. A. Shamma, W. Maass. *Integration of stimulus history in information conveyed by neurons in primary auditory cortex in response to tone sequences.* Program No. 163.8. 2009 Neuroscience Meeting Planner. Chicago, IL: Society for Neuroscience, 2009. Online.

5. S. Klampfl, W. Maass. *Replacing supervised classification learning by Slow Feature Analysis in spiking neural networks.* In Proc. of NIPS 2009, Advances in Neural Information Processing Systems, volume 22, pp. 988-996. MIT Press, 2010.

6. S. Klampfl, W. Maass. *A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction.* Neural Computation, 22(12):2979-3035, 2010.

7. S. Klampfl, S. V. David, P. Yin, S. A. Shamma, W. Maass. *Neurons in primary auditory cortex integrate information about past and present stimuli.* Submitted for publication. 2011.

A.1 Comments and Contributions to Publications

The first publication, *Extracting statistically independent components with a generalized BCM rule for spiking neurons*, is my Master's thesis. The results are not included in this thesis, however, certain aspects of it provide the basis for the next two publications.

The paper *Information bottleneck optimization and independent component extraction with spiking neurons* was written by myself (SK), Robert Legenstein (RL),

and my supervisor Wolfgang Maass (WM). The derivation of the spike based learning rule and its analysis was performed by SK. The simplified rate based rule was developed by RL and analyzed by both RL and SK. The simulation experiments were designed and conducted by SK. SK wrote the paper, with additional input from WM and RL. The paper was selected for a *poster* presentation at the *20th Annual Conference on Neural Information Processing Systems* (NIPS) 2006, Vancouver, Canada. This approach was extended in the journal paper *Spiking neurons can learn to solve information bottleneck problems and to extract independent components* by the same authors, published in *Neural Computation*. This publication includes a more detailed analysis of the learning rules as well as a number of additional simulation experiments. This paper provides the basis for Chapter 2 of this thesis.

The paper *Replacing supervised classification learning by Slow Feature Analysis in spiking neural networks* was written by myself (SK) and my supervisor Wolfgang Maass (WM). The derivation and analysis of the relationship between Slow Feature Analysis and Fisher's Linear Discriminant was performed by SK. The simulation experiments were designed by SK and WM and conducted by SK. The paper was written by SK, with additional input from WM. The paper was selected for a *spotlight* presentation at the *23rd Annual Conference on Neural Information Processing Systems* (NIPS), 2009, Vancouver, Canada. The ideas of this publication were extended in the article *A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction* by the same authors, published in *Neural Computation*. This publication includes a more detailed derivation and analysis of the main results as well as a number of additional simulation experiments. This paper provides the basis for Chapter 3 of this thesis.

The poster *Integration of stimulus history in information conveyed by neurons in primary auditory cortex in response to tone sequences* was presented at the *Annual Meeting 2009 of the Society for Neuroscience*, Chicago, IL, USA. It resulted from a joint work with Stephen V. David (SVD), Pingbo Yin (PY), and Shihab A. Shamma (SAS) from the Institute for Systems Research, University of Maryland, as well as my supervisor Wolfgang Maass (WM). The results presented there were extended into the paper *Neurons in primary auditory cortex integrate information about past and present stimuli* by the same authors. It has been submitted for publication. The biological experiments were designed by SVD, PY, and SAS, and conducted by SVD ad PY. The data was analyzed by SK; SVD and WM provided additional ideas for the data analysis. The paper was written by SK, with additional input from SVD and WM. This paper is the basis for Chapter 4 of this thesis.

Appendix B

Appendix to Chapter 2: Spiking neurons can learn to solve information bottleneck problems and to extract independent components

Contents

B.1 Details of the derivations of the learning rules

B.1.1 Evaluation of firing and joint firing probabilities

To quantify the information between output spike trains $\mathbf{Y}_1^K$ and $\mathbf{Y}_2^K$ of length $K\Delta t$ we need an expression for the joint probability $P(Y_1^K, Y_2^K)$. For given input spike trains $X^k = (X_1^k, \ldots, X_N^k)$ up to time step k and postsynaptic spike history Y_i^{k-1} we can write the probability of emitting a postsynaptic spike in the k-th time step using the firing probability ρ_i^k (2.2) as the binary distribution

$$P(y_i^k | Y_i^{k-1}, X^k) = (\rho_i^k)^{y_i^k} (1 - \rho_i^k)^{(1-y_i^k)}. \tag{B.1}$$

The marginal probability, given only the postsynaptic history, can be written as

$$P(y_i^k | Y_i^{k-1}) = (\bar{\rho}_i^k)^{y_i^k} (1 - \bar{\rho}_i^k)^{(1-y_i^k)}, \tag{B.2}$$

where $\bar{\rho}_i^k = \langle \rho_i^k \rangle_{\mathbf{X}^k | Y_i^{k-1}} = \sum_{X^k} \rho_i^k P(X^k | Y_i^{k-1})$ is the average firing probability in the k-th time step (where ρ_i^k depends of course on X^k and Y_i^k). The probability of an entire output spike train Y_i^K given the input X^K is then obtained by

$$P(Y_i^K | X^K) = \prod_{k=1}^{K} P(y_i^k | Y_i^{k-1}, X^k) \tag{B.3}$$

and analogously, the probability of an output spike train by

$$P(Y_i^K) = \prod_{k=1}^{K} P(y_i^k | Y_i^{k-1}). \tag{B.4}$$

If two neurons receive the same input at their synapses and produce outputs Y_1^K and Y_2^K, we can write the joint probability of spiking in the k-th time step given the postsynaptic histories and the input as

$$P(y_1^k, y_2^k | Y_1^{k-1}, Y_2^{k-1}, X^k) = P(y_1^k | Y_1^{k-1}, X^k) P(y_2^k | Y_2^{k-1}, X^k). \tag{B.5}$$

The marginal probability given only the postsynaptic histories can be written using (B.1) as

$$\begin{aligned} &P(y_1^k, y_2^k | Y_1^{k-1}, Y_2^{k-1}) = \\ &\qquad \left\langle (\rho_1^k)^{y_1^k} (1-\rho_1^k)^{(1-y_1^k)} (\rho_2^k)^{y_2^k} (1-\rho_2^k)^{(1-y_2^k)} \right\rangle_{\mathbf{X}^k | Y_1^{k-1}, Y_2^{k-1}} = \\ &(\bar{\bar{\rho}}_{12}^k)^{y_1^k y_2^k} (\bar{\bar{\rho}}_1^k - \bar{\bar{\rho}}_{12}^k))^{y_1^k (1-y_2^k)} (\bar{\bar{\rho}}_2^k - \bar{\bar{\rho}}_{12}^k)^{(1-y_1^k) y_2^k} (1 - \bar{\bar{\rho}}_1^k - \bar{\bar{\rho}}_2^k + \bar{\bar{\rho}}_{12}^k)^{(1-y_1^k)(1-y_2^k)}, \end{aligned} \tag{B.6}$$

where $\bar{\bar{\rho}}_i^k = \left\langle \rho_i^k \right\rangle_{\mathbf{X}^k | Y_1^{k-1}, Y_2^{k-1}} = \sum_{X^k} \rho_i^k P(X^k | Y_1^{k-1}, Y_2^{k-1})$ is the average firing probability of neuron i, given the postsynaptic history of both neurons, and $\bar{\bar{\rho}}_{12}^k = \left\langle \rho_1^k \rho_2^k \right\rangle_{\mathbf{X}^k | Y_1^{k-1}, Y_2^{k-1}}$ is the average product of firing probabilities of both neurons. The joint probability of two entire output spike trains is then finally given as

$$P(Y_1^K, Y_2^K) = \prod_{k=1}^{K} P(y_1^k, y_2^k | Y_1^{k-1}, Y_2^{k-1}). \tag{B.7}$$

B.1.2 Evaluation of the gradient of ΔL_{12}^k

We have to calculate the gradient $\partial \Delta L_{12}^k / \partial w_{1j}$, with

$$\Delta L_{12}^k = \left\langle \beta \log \frac{P(y_1^k, y_2^k | Y_1^{k-1}, Y_2^{k-1})}{P(y_1^k | Y_1^{k-1}) P(y_2^k | Y_2^{k-1})} \right\rangle_{\mathbf{X}^k, \mathbf{Y}_1^k, \mathbf{Y}_2^k}. \tag{B.8}$$

The strategy for the derivation is similar as for the terms considered in (Toyoizumi et al., 2005), but a number of details are different.

We treat the nominator and the two product terms of the denominator in (B.8) separately. The average of an arbitrary function f_w with arguments x, y_1 and y_2 is by definition

$$\begin{aligned} \langle f_w(x, y_1, y_2) \rangle_{\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2} &= \sum_{x, y_1, y_2} p_w(x, y_1, y_2) f_w(x, y_1, y_2) \\ &= \left\langle \sum_{y_1} p_w(y_1 | x) f_w(x, y_1, y_2) \right\rangle_{\mathbf{x}, \mathbf{y}_2}, \end{aligned} \tag{B.9}$$

where $p_w(x, y_1, y_2) = p(x)p(y_2|x)p_w(y_1|x)$ denotes the joint probability of the triple (x, y_1, y_2) to occur, assuming that $\mathbf{y}_1$ is independent of $\mathbf{y}_2$ given $\mathbf{x}$. The subscript w indicates that both the probability distribution p_w and the function f_w depend on an additional parameter w.

Taking the derivative with respect to w, the product rule yields two terms,

$$\begin{aligned}\frac{\partial}{\partial w}\left\langle f_w(x, y_1, y_2)\right\rangle_{\mathbf{x},\mathbf{y}_1,\mathbf{y}_2} = &\left\langle \sum_{y_1} p_w(y_1|x)\frac{\partial}{\partial w} f_w(x, y_1, y_2)\right\rangle_{\mathbf{x},\mathbf{y}_2} \\ &+ \left\langle \sum_{y_1} \frac{\partial}{\partial w} p_w(y_1|x) f_w(x, y_1, y_2)\right\rangle_{\mathbf{x},\mathbf{y}_2},\end{aligned} \tag{B.10}$$

where the first term contains the derivative of the function f_w and the second term contains the derivative of the conditional probability p_w. Since

$$\frac{\partial}{\partial w} p_w(y_1|x) = p_w(y_1|x)\frac{\partial}{\partial w}\log p_w(y_1|x), \tag{B.11}$$

the right-hand side of (B.10) evaluates to

$$\left\langle \frac{\partial}{\partial w} f_w(x, y_1, y_2)\right\rangle_{\mathbf{x},\mathbf{y}_1,\mathbf{y}_2} + \left\langle \left[\frac{\partial}{\partial w}\log p_w(y_1|x)\right] f_w(x, y_1, y_2)\right\rangle_{\mathbf{x},\mathbf{y}_1,\mathbf{y}_2}, \tag{B.12}$$

i.e., it can be written as an average over the joint distribution of x, y_1 and y_2.

Now we can evaluate each of the terms of (B.8) using (B.12). Considering the term $\frac{\partial}{\partial w_{1j}}\left\langle \log P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right\rangle_{\mathbf{X}^k,\mathbf{Y}_1^k,\mathbf{Y}_2^k}$ first, we get

$$\begin{aligned}&\left\langle \frac{\partial}{\partial w_{1j}}\log P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right\rangle_{\mathbf{X}^k,\mathbf{Y}_1^k,\mathbf{Y}_2^k} \\ &\qquad + \left\langle \left[\frac{\partial}{\partial w_{1j}}\log P(Y_2^k|X^k)\right]\log P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right\rangle_{\mathbf{X}^k,\mathbf{Y}_1^k,\mathbf{Y}_2^k}.\end{aligned} \tag{B.13}$$

We find that the first term of (B.13) vanishes because

$$\begin{aligned}&\left\langle \frac{\partial}{\partial w_{1j}}\log P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right\rangle_{\mathbf{X}^k,\mathbf{Y}_1^k,\mathbf{Y}_2^k} = \\ &= \left\langle \left\langle \frac{\partial}{\partial w_{1j}}\log P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right\rangle_{\mathbf{y}_1^k,\mathbf{y}_2^k|Y_1^{k-1},Y_2^{k-1}}\right\rangle_{\mathbf{Y}_1^{k-1},\mathbf{Y}_2^{k-1}} \\ &= \left\langle \sum_{y_1^k,y_2^k}\left[\frac{\partial}{\partial w_{1j}}\log P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right] P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right\rangle_{\mathbf{Y}_1^{k-1},\mathbf{Y}_2^{k-1}} \\ &= \left\langle \frac{\partial}{\partial w_{1j}}\left[\sum_{y_1^k,y_2^k} P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})\right]\right\rangle_{\mathbf{Y}_1^{k-1},\mathbf{Y}_2^{k-1}} = 0.\end{aligned} \tag{B.14}$$

In the second line of (B.14) we drop the expectation over $\mathbf{X}^k$ since the argument of the expectation operator is independent of the input spike train X^k and use the identity $\langle\cdot\rangle_{\mathbf{Y}_1^k,\mathbf{Y}_2^k} = \left\langle\rangle\cdot\right\rangle_{\mathbf{y}_1^k,\mathbf{y}_2^k|Y_1^{k-1},Y_2^{k-1}}\right\rangle_{\mathbf{Y}_1^{k-1},\mathbf{Y}_2^{k-1}}$. With the same argument it can be shown that

$$\left\langle \frac{\partial}{\partial w_{1j}} \log P(y_i^k|Y_i^{k-1}) \right\rangle_{\mathbf{X}^k,\mathbf{Y}_1^k,\mathbf{Y}_2^k} = \left\langle \frac{\partial}{\partial w_{1j}} \log P(y_i^k|Y_i^{k-1},X^k) \right\rangle_{\mathbf{X}^k,\mathbf{Y}_1^k,\mathbf{Y}_2^k} = 0 \tag{B.15}$$

for $i = 1, 2$. Hence, the only term that gives a nontrivial contribution in (B.13) is the second one. With an analogous evaluation for the other terms in (B.8) we finally have

$$\frac{\partial}{\partial w_{1j}} \Delta L_{12}^k = \left\langle \left[\frac{\partial}{\partial w_{1j}} \log P(Y_1^k|X^k) \right] \log \frac{P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})}{P(y_1^k|Y_1^{k-1})P(y_2^k|Y_2^{k-1})} \right\rangle_{\mathbf{X}^k,\mathbf{Y}_1^k,\mathbf{Y}_2^k}. \tag{B.16}$$

Now we can identify the factors

$$C_{1j}^k := \frac{\partial}{\partial w_{1j}} \log P(Y_1^k|X^k) = \sum_{l=1}^{k} \left[\frac{y_1^l}{\rho_1^l} - \frac{1-y_1^l}{1-\rho_1^l} \right] \frac{\partial \rho_1^l}{\partial u_1} \sum_{n=1}^{l} \varepsilon(t^l - t^n) x_j^n \tag{B.17}$$

and

$$\begin{aligned} F_{12}^k &:= \log \frac{P(y_1^k, y_2^k|Y_1^{k-1}, Y_2^{k-1})}{P(y_1^k|Y_1^{k-1})P(y_2^k|Y_2^{k-1})} \\ &= y_1^k y_2^k \log \frac{\bar{\bar{\rho}}_{12}^k}{\bar{\rho}_1^k \bar{\rho}_2^k} + y_1^k (1-y_2^k) \log \frac{\bar{\bar{\rho}}_1^k - \bar{\bar{\rho}}_{12}^k}{\bar{\rho}_1^k - \bar{\rho}_1^k \bar{\rho}_2^k} + \\ &\quad + (1-y_1^k) y_2^k \log \frac{\bar{\bar{\rho}}_2^k - \bar{\bar{\rho}}_{12}^k}{\bar{\rho}_2^k - \bar{\rho}_1^k \bar{\rho}_2^k} + (1-y_1^k)(1-y_2^k) \log \frac{1 - \bar{\bar{\rho}}_1^k - \bar{\bar{\rho}}_2^k + \bar{\bar{\rho}}_{12}^k}{1 - \bar{\rho}_1^k - \bar{\rho}_1^k + \bar{\rho}_1^k \bar{\rho}_2^k}. \end{aligned} \tag{B.18}$$

For computational reasons we approximate the sum $\sum_{l=1}^k$ in the correlation term C_{1j}^k (B.17) by an exponential window with time constant $\tau_C = 1$s (Toyoizumi et al., 2005):

$$C_{1j}^k = C_{1j}^{k-1} \left(1 - \frac{\Delta t}{\tau_C} \right) + \sum_{n=1}^{k} \varepsilon(t^k - t^n) x_j^n \frac{g'(u_1(t^k))}{g(u_1(t^k))} \left[y_1^k - \rho_1^k \right]. \tag{B.19}$$

Furthermore, if we make the assumption $\bar{\bar{\rho}}_i^k = \bar{\rho}_i^k$ (see appendix B.1.3) we can simplify the term F_{12}^k (B.18) and write $\bar{\rho}_i^k = \bar{g}_i(t^k) R_i(t^k) \Delta t$ and $\bar{\bar{\rho}}_{12}^k = \bar{g}_{12}(t^k) R_1(t^k)\ R_2(t^k) (\Delta t)^2$ with $\bar{g}_i(t^k) = \langle g(u_i(t^k)) \rangle_{\mathbf{X}^k|Y_i^{k-1}}$ and $\bar{g}_{12}(t^k) = \langle g(u_1(t^k))\ g(u_2(t^k)) \rangle_{\mathbf{X}^k|Y_1^{k-1},Y_2^{k-1}}$. Using the approximation $\log(1-x) \approx -x$ for small x we

get

$$F_{12}^k = y_1^k y_2^k \log \frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)\bar{g}_2(t^k)} - y_1^k(1-y_2^k)R_2(t^k)\Delta t \left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)} - \bar{g}_2(t^k)\right] - \\ - (1-y_1^k)y_2^k R_1(t^k)\Delta t \left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)} - \bar{g}_1(t^k)\right] + \\ + (1-y_1^k)(1-y_2^k)R_1(t^k)R_2(t^k)(\Delta t)^2 \left[\bar{g}_{12}(t^k) - \bar{g}_1(t^k)\bar{g}_2(t^k)\right]. \tag{B.20}$$

This approximation is valid for small Δt.

The weight change is then finally given by

$$\Delta\tilde{w}_{1j}^k = \alpha \left\langle C_{1j}^k \beta F_{12}^k \right\rangle_{\mathbf{X}^k, \mathbf{Y}_1^k, \mathbf{Y}_2^k}. \tag{B.21}$$

B.1.3 A closer look at the firing probabilities $\bar{\bar{\rho}}_i^k$ and $\bar{\rho}_i^k$

For simplicity we assume $i = 1$. Using equations (B.1) and (B.2) we can write for ρ_1^k and $\bar{\rho}_1^k$:

$$\rho_1^k = P(y_1^k = 1 | X^k, Y_1^{k-1}), \text{ and} \tag{B.22}$$
$$\bar{\rho}_1^k = P(y_1^k = 1 | Y_1^{k-1}). \tag{B.23}$$

From $\bar{\bar{\rho}}_1^k = \langle \rho_1^k \rangle_{\mathbf{X}^k | Y_1^{k-1}, Y_2^{k-1}}$ we find that

$$\bar{\bar{\rho}}_1^k = P(y_1^k = 1 | Y_1^{k-1}, Y_2^{k-1}). \tag{B.24}$$

Qualitatively, any difference between $\bar{\rho}_1^k$ and $\bar{\bar{\rho}}_1^k$ arises from the additional information that, given the postsynaptic history Y_1^{k-1}, the output of the other neuron, Y_2^{k-1}, conveys about a postsynaptic event at time step k. For a learning rule that uses the term F_{12}^k (2.17) we have to calculate $\bar{\bar{\rho}}_i^k$ online. The average firing probabilities $\bar{\rho}_i^k = \langle \rho_i^k \rangle_{\mathbf{X}^k | Y_i^{k-1}}$ are implemented as running averages of ρ_i^k, as in (Toyoizumi et al., 2005).

We can express $\bar{\bar{\rho}}_1^k$ (B.24) using $\bar{\rho}_1^k$ (B.23), i.e.,

$$\bar{\bar{\rho}}_1^k = \bar{\rho}_1^k \cdot \frac{P(Y_2^{k-1} | y_1^k = 1, Y_1^{k-1})}{P(Y_2^{k-1} | Y_1^{k-1})}. \tag{B.25}$$

The second factor in (B.25) is hard to evaluate online. However, if we assume that $y_1^k = 1$ is independent from Y_2^{k-1} given Y_1^{k-1}, i.e., that $P(y_1^k = 1, Y_2^{k-1} | Y_1^{k-1}) = P(y_1^k = 1 | Y_1^{k-1}) P(Y_2^{k-1} | Y_1^{k-1})$, we can set $\bar{\bar{\rho}}_1^k = \bar{\rho}_1^k$. In this case, since

$$\bar{\rho}_1^k = \left\langle \bar{\bar{\rho}}_1^k \right\rangle_{\mathbf{Y}_2^{k-1} | Y_1^{k-1}}, \tag{B.26}$$

we replace $\bar{\bar{\rho}}_1^k$ by its mean value with respect to the distribution $P(Y_2^{k-1} | Y_1^{k-1})$.

B.1.4 Derivation of the simplified learning rule

The starting point for the derivation for this simplified model is the weight update rule (2.18)

$$\begin{aligned}\frac{\Delta w_{1j}^k}{\Delta t} &= -\alpha \left\langle C_{1j}^k B_1^k(-\gamma) + \alpha\beta\Delta t C_{1j}^k B_{12}^k \right\rangle_{\mathbf{Y}_1^k,\mathbf{Y}_2^k,\mathbf{X}^k} \\ &= -\alpha \left\langle \left\langle C_{1j}^k B_1(-\gamma) \right\rangle_{\mathbf{Y}_1^k|X^k} + \alpha\beta\Delta t \left\langle C_{1j}^k B_{12}^k \right\rangle_{\mathbf{Y}_1^k,\mathbf{Y}_2^k|X^k} \right\rangle_{\mathbf{X}^k}.\end{aligned} \tag{B.27}$$

where in contrast to (2.18) we consider the batch version of the learning rule in which the weight update is averaged over the input and output distribution.

For notational convenience, we write the correlation term (B.17) as[1]:

$$\begin{aligned}C_{1j}^k &= \sum_{l=1}^{k} \left[\frac{y_1^l}{\rho_1^l} - \frac{1-y_1^l}{1-\rho_1^l}\right] \frac{\partial \rho_1^l}{\partial u_1} \sum_{n=1}^{l} \varepsilon(t^l - t^n) x_j^n \\ &= \sum_{l=1}^{k} \left[y_1^l - \rho_1^l\right] \frac{(\rho_1^l)'}{\rho_1^l(1-\rho_1^l)} \sum_{n=1}^{l} \varepsilon(t^l - t^n) x_j^n \\ &= \sum_{l=1}^{k} K_1(l) \left[y_1^l - \rho_1^l\right],\end{aligned}$$

with

$$K_1(l) = \frac{(\rho_1^l)'}{\rho_1^l(1-\rho_1^l)} \sum_{n=1}^{l} \varepsilon(t^l - t^n) x_j^n \approx \frac{g'(u_1(t^l))}{g(u_1(t^l))} \sum_{n=1}^{l} \varepsilon(t^l - t^n) x_j^n.$$

Here, we used the approximation $\rho_1^l \approx g(u_1(t^l))\Delta t$, which holds for small Δt. Furthermore, we write the postsynaptic term as

$$B_1^k(-\gamma) = \frac{y_1^k}{\Delta t} B_{1A}^k + (1 - y_1^k) B_{1B}^k, \tag{B.28}$$

with

$$\begin{aligned}B_{1A}^k &= \log\left[\frac{g(u_1(t^k))}{\bar{g}_1(t^k)} \left(\frac{\bar{g}_1(t^k)}{\tilde{g}}\right)^{\gamma}\right], \\ B_{1B}^k &= -R_1(t^k)[g(u_1(t^k)) - (1-\gamma)\bar{g}_1(t^k) - \gamma\tilde{g}].\end{aligned}$$

Since $\langle y_i^k \rangle_{\mathbf{Y}_i^k|X^k} = \rho_i^k$ and $\langle y_i^l y_i^k \rangle_{\mathbf{Y}_i^k|X^k} = \delta_{lk}\rho_i^k + \rho_i^l \rho_i^k (1 - \delta_{lk})$, where δ_{lk} is the Kronecker delta function, we get

$$\begin{aligned}\langle C_{1j}^k B_1^k(-\gamma) \rangle_{\mathbf{Y}_1^k|X^k} &= \left\langle \sum_{l=1}^{k} K_1(l)[y_1^l - \rho_1^l] \left(\frac{y_1^k}{\Delta t} B_{1A}^k + (1 - y_1^k) B_{1B}^k\right) \right\rangle_{\mathbf{Y}_1^k|X^k} \\ &= \sum_{l=1}^{k} K_1(l)\delta_{lk} \left[\frac{\rho_1^k}{\Delta t} B_{1A}^k - \frac{(\rho_1^k)^2}{\Delta t} B_{1A}^k - \rho_1^k B_{1B}^k + (\rho_1^k)^2 B_{1B}^k\right].\end{aligned} \tag{B.29}$$

[1]For simplicity, we write $g(u)$ instead of $g_{alt}(u)$ throughout this section.

Since $\rho_i^k \approx g(u_i(t^k))\Delta t$, we get

$$\begin{aligned}\langle C_{1j}^k B_1^k(-\gamma)\rangle_{\mathbf{Y}_1^k|X^k} &= K_1(k)\Big[g(u_1(t^k))B_{1A}^k - g(u_1(t^k))^2\Delta t B_{1A}^k \\ &\quad - g(u_1(t^k))\Delta t B_{1B}^k + g(u_1(t^k))^2(\Delta t)^2 B_{1B}^k\Big] \\ &= K_1(k)\Big[g(u_1(t^k))B_{1A}^k + O(\Delta t)\Big] \\ &\approx K_1(k)g(u_1(t^k))B_{1A}^k,\end{aligned} \tag{B.30}$$

where we assume small Δt. Substitution of $K_1(k)$ and B_{1A}^k yields

$$\langle C_{1j}^k B_1^k(-\gamma)\rangle_{\mathbf{Y}_1^k|X^k} \approx \nu_j^{pre,k} f(\nu_1^k)\log\left[\frac{\nu_1^k}{\bar{\nu}_1^k}\left(\frac{\bar{\nu}_1^k}{\tilde{g}}\right)^\gamma\right], \tag{B.31}$$

where the presynaptic rate at synapse j is denoted by $\nu_j^{pre,k} = a\sum_{n=1}^k \varepsilon(t^k - t^n)x_j^n$ with a in units $(\mathrm{Vs})^{-1}$, and $\bar{\nu}_1^k$, $\bar{\nu}_2^k$, $\bar{\nu}_{12}^k$ are running averages of the output rate ν_1^k, the target rate ν_2^k, and the product of these values, $\nu_1^k\nu_2^k$. The rate ν_1^k is given directly by $g_{alt}(u_1(t^k))$. The function $f(\nu_1^k) = g'(g^{-1}(\nu_1^k))/a$ is proportional to the derivative of g with respect to u, evaluated at the current membrane potential.

For the evaluation of the second term in (B.27), we write B_{12}^k as

$$B_{12}^k = \frac{y_1^k y_2^k}{(\Delta t)^2}D_{12}^k - \frac{y_1^k(1-y_2^k)}{\Delta t}D_1^k - \frac{(1-y_1^k)y_2^k}{\Delta t}D_2^k + (1-y_1^k)(1-y_2^k)D_0, \tag{B.32}$$

with

$$\begin{aligned}D_{12}^k &= \log\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)\bar{g}_2(t^k)}, \\ D_1^k &= \frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)} - \bar{g}_2(t^k), \\ D_2^k &= \frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)} - \bar{g}_1(t^k), \\ D_0^k &= \bar{g}_{12}(t^k) - \bar{g}_1(t^k)\bar{g}_2(t^k).\end{aligned}$$

We get

$$\begin{aligned}\Big\langle C_{1j}^k B_{12}^k\Big\rangle_{\mathbf{Y}_1^k,\mathbf{Y}_2^k|X^k} = \sum_{l=1}^k K_1(l)\Big\langle &\Big[y_1^l - \rho_1^l\Big]\Big[\frac{y_1^k y_2^k}{(\Delta t)^2}D_{12}^k - \\ &- \frac{y_1^k}{\Delta t}D_1^k + \frac{y_1^k y_2^k}{\Delta t}D_1^k - \frac{y_2^k}{\Delta t}D_2^k + \frac{y_1^k y_2^k}{\Delta t}D_2^k - \\ &- D_0^k + y_1^k D_0^k + y_2^k D_0^k - y_1^k y_2^k D_0^k\Big]\Big\rangle_{\mathbf{Y}_1^k,\mathbf{Y}_2^k|X^k}.\end{aligned}$$

For given input X^k, the two spike trains Y_1^k and Y_2^k are independent and $\langle y_1^l y_2^k\rangle_{\mathbf{Y}_1^k,\mathbf{Y}_2^k|X^k} = \rho_1^l\rho_2^k$. Furthermore, we use $\langle y_1^l y_1^k y_2^k\rangle_{\mathbf{Y}_1^k,\mathbf{Y}_2^k|X^k} = \langle y_1^l y_1^k\rangle_{\mathbf{Y}_1^k|X^k}\rho_2^k$,

to get

$$\begin{aligned}\left\langle C_{1j}^k B_{12}^k \right\rangle_{\mathbf{Y}_1^k, \mathbf{Y}_2^k | X^k} &= K_1(k) g(u_1(t^k))[g(u_1(t^k)) D_{12}^k - D_1^k + O(\Delta t)] \\ &\approx K_1(k) g(u_1(t^k))[g(u_2(t^k)) D_{12}^k - D_1^k] \\ &= \nu_j^{pre}(t^k) f(\nu_1(t^k)) \left[\nu_2^k \log \frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k} - \left(\frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k} - \bar{\nu}_2^k \right) \right].\end{aligned} \tag{B.33}$$

Again, the approximation is valid for small Δt.

Substitution of (B.31) and (B.33) into (B.27) yields

$$\begin{aligned}\frac{\Delta w_{1j}^k}{\Delta t} = -\alpha \nu_j^{pre,k} f(\nu_1^k) \Bigg\{ &\log \left[\frac{\nu_1^k}{\bar{\nu}_1^k} \left(\frac{\bar{\nu}_1^k}{\tilde{g}} \right)^\gamma \right] \\ &- \beta \Delta t \left(\nu_2^k \log \left[\frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k} \right] - \bar{\nu}_2^k \left[\frac{\bar{\nu}_{12}^k}{\bar{\nu}_1^k \bar{\nu}_2^k} - 1 \right] \right) \Bigg\},\end{aligned} \tag{B.34}$$

where the expectation $\langle \cdot \rangle_{\mathbf{X}}$ in (B.27) is approximated by averaging over a single long trial under the assumption of a small learning rate α.

B.2 Whitening transform

For the whitening transform used in the experiment described in section 2.5.5 we define a vector $\mathbf{x}(t) = [r_1(t) - \langle r_1 \rangle, r_2(t) - \langle r_2 \rangle]^T$, where $r_1(t)$ and $r_2(t)$ are the signals which should be whitened (the rate modulations of one input and the target signal in this case). Note that the averages of both signals are subtracted as to make $\mathbf{x}$ have zero mean. Furthermore, let $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$ denote the 2-by-2 covariance matrix of $\mathbf{x}$, which is calculated in the simulations as the empirical covariance matrix of 10s-samples of $r_1(t)$ and $r_2(t)$. The whitening transform is then given by

$$\mathbf{T} = \mathbf{E}\mathbf{D}^{-\frac{1}{2}}\mathbf{E}^T, \tag{B.35}$$

where $\mathbf{E}$ is the orthogonal matrix of eigenvectors of $\mathbf{C}$ and $\mathbf{D}$ is the diagonal matrix of its eigenvalues, $\mathbf{D} = \mathrm{diag}(\lambda_1, \lambda_2)$ (i.e., $\mathbf{C} = \mathbf{E}\mathbf{D}\mathbf{E}^T$.). With this transformation the vectors $\mathbf{T}\mathbf{x}$ have unit variance; in order to scale them back to the variance of the original signals we define an additional scaling matrix $\mathbf{S} = \mathrm{diag}(\sigma_1, \sigma_2)$, where σ_1 and σ_2 are the standard deviations of r_1 and r_2, respectively. With the means added back, which have been subtracted before, the total transformation is then given by

$$\tilde{\mathbf{x}} = \mathbf{S}\mathbf{T}\mathbf{x} + \begin{bmatrix} \langle r_1 \rangle \\ \langle r_2 \rangle \end{bmatrix}, \tag{B.36}$$

where the elements of $\tilde{\mathbf{x}}(t) = [\tilde{r}_1(t), \tilde{r}_2(t)]^T$ are uncorrelated.

B.3 Derivation of the approximation in section 2.6.1

Remember that the combined postsynaptic term of the learning rule of neuron i (2.32) can be written as

$$A_i^k := B_i^k(\gamma) - \beta \Delta t B_{12}^k, \tag{B.37}$$

where

$$B_i^k(\gamma) = \frac{y_i^k}{\Delta t}\log\left[\frac{g(u_i(t^k))}{\bar{g}_i(t^k)}\left(\frac{\tilde{g}}{\bar{g}_i(t^k)}\right)^\gamma\right] - (1-y_i^k)R_i(t^k)\left[g(u_i(t^k)) - (1+\gamma)\bar{g}_i(t^k) + \gamma\tilde{g}\right], \tag{B.38}$$

and

$$\begin{aligned} B_{12}^k = {} & \frac{y_1^k y_2^k}{(\Delta t)^2}\log\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)\bar{g}_2(t^k)} - \frac{y_1^k}{\Delta t}(1-y_2^k)R_2(t^k)\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)} - \bar{g}_2(t^k)\right] \\ & - \frac{y_2^k}{\Delta t}(1-y_1^k)R_1(t^k)\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)} - \bar{g}_1(t^k)\right] \\ & + (1-y_1^k)(1-y_2^k)R_1(t^k)R_2(t^k)\left[\bar{g}_{12}(t^k) - \bar{g}_1(t^k)\bar{g}_2(t^k)\right]. \end{aligned} \tag{B.39}$$

For simplicity we consider only neuron 1 in the following; symmetric arguments apply for the case of neuron 2. We can distinguish between 4 postsynaptic states for both neurons in each time step k: one where both are spiking ($y_1^k = y_2^k = 1$), one where neither of them emits a spike ($y_1^k = y_2^k = 0$) and two cases where only one of them fires ($y_1^k = 1$, $y_2^k = 0$, and $y_1^k = 0$, $y_2^k = 1$, respectively). For these four cases the postsynaptic term (B.37) evaluates to

- $y_1^k = y_2^k = 1$:

$$A_1^k = \frac{1}{\Delta t}\log\left[\frac{g(u_1(t^k))}{\bar{g}_1(t^k)}\left(\frac{\tilde{g}}{\bar{g}_1(t^k)}\right)^\gamma\right] - \frac{1}{\Delta t}\beta\log\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)\bar{g}_2(t^k)}, \tag{B.40}$$

- $y_1^k = 1$, $y_2^k = 0$:

$$A_1^k = \frac{1}{\Delta t}\log\left[\frac{g(u_1(t^k))}{\bar{g}_1(t^k)}\left(\frac{\tilde{g}}{\bar{g}_1(t^k)}\right)^\gamma\right] + \beta R_2(t^k)\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)} - \bar{g}_2(t^k)\right], \tag{B.41}$$

- $y_1^k = 0$, $y_2^k = 1$:

$$A_1^k = -R_1(t^k)\left[g(u_1(t^k)) - (1+\gamma)\bar{g}_1(t^k) + \gamma\tilde{g}\right] + \beta R_1(t^k)\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)} - \bar{g}_1(t^k)\right], \tag{B.42}$$

- $y_1^k = y_2^k = 0$:

$$A_1^k = -R_1(t^k)\left[g(u_1(t^k)) - (1+\gamma)\bar{g}_1(t^k) + \gamma\tilde{g}\right] - \beta\Delta t R_1(t^k)R_2(t^k)\left[\bar{g}_{12}(t^k) - \bar{g}_1(t^k)\bar{g}_2(t^k)\right]. \tag{B.43}$$

We want to model the contribution of the term B_{12}^k (B.39) by changing the value $g(u_1(t^k))$. That is, we again apply the simple postsynaptic BCM-term,

$$\begin{aligned}\hat{B}_1^k(\gamma) = \frac{y_1^k}{\Delta t}\log\left[\frac{\hat{g}_1(t^k)}{\bar{g}_1(t^k)}\left(\frac{\tilde{g}}{\bar{g}_1(t^k)}\right)^{\gamma}\right] \\ -(1-y_1^k)R_1(t^k)\left[\hat{g}_1(t^k)-(1+\gamma)\bar{g}_1(t^k)+\gamma\tilde{g}\right],\end{aligned} \tag{B.44}$$

instead of the combined postsynaptic term A_1^k (B.37) in the learning rule of neuron 1, but encapsulate the effect of the term B_{12}^k in changing the gain $g(u_1(t^k))$ into $\hat{g}_1(t^k)$ in this simple postsynaptic term $\hat{B}_1^k$.

We look for arithmetic expressions for $\hat{g}_1(t^k)$ by comparing formula (B.44) with equations (B.40) to (B.43). We get

- $y_1^k = y_2^k = 1$:

$$\hat{g}_1(t^k) = g(u_1(t^k))\left(\frac{\bar{g}_1(t^k)\bar{g}_2(t^k)}{\bar{g}_{12}(t^k)}\right)^{\beta}, \tag{B.45}$$

- $y_1^k = 1$, $y_2^k = 0$:

$$\hat{g}_1(t^k) = g(u_1(t^k))\exp\left[R_2(t^k)\beta\Delta t\left(\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)}-\bar{g}_2(t^k)\right)\right], \tag{B.46}$$

- $y_1^k = 0$, $y_2^k = 1$:

$$\hat{g}_1(t^k) = g(u_1(t^k)) - \beta\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)}-\bar{g}_1(t^k)\right], \tag{B.47}$$

- $y_1^k = y_2^k = 0$:

$$\hat{g}_1(t^k) = g(u_1(t^k)) + R_2(t^k)\beta\Delta t\left[\bar{g}_{12}(t^k)-\bar{g}_1(t^k)\bar{g}_2(t^k)\right]. \tag{B.48}$$

However, Figure 2.3 suggests that significant effects of B_{12}^k are encountered only when one of the two neurons is firing; we also neglect the influence of simultaneous action potentials within the same time step as Δt gets small. Therefore we focus only on cases (B.46) and (B.47) where exactly one of the two neurons is firing. The value $g(u_1(t^k))$ is then modified according to

$$\hat{g}_1(t^k) = g(u_1(t^k))\exp\left[R_2(t^k)\beta\Delta t\left(\frac{\bar{g}_{12}(t^k)}{\bar{g}_1(t^k)}-\bar{g}_2(t^k)\right)\right] \quad \text{if } y_1^k = 1,\ y_2^k = 0, \tag{B.49}$$

which corresponds to a multiplicative change, and

$$\hat{g}_1(t^k) = g(u_1(t^k)) - \tilde{\beta}\left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_2(t^k)}-\bar{g}_1(t^k)\right] \quad \text{if } y_2^k = 1,\ y_1^k = 0, \tag{B.50}$$

which corresponds to an additive change.

Summarizing, the modified value $\hat{g}_i(t^k)$ for neuron $i = 1, 2$ can be written as follows:

$$\hat{g}_i(t^k) = g(u_i(t^k)) \cdot a_i(t^k)^{y_i^k(1-y_{3-i}^k)} + b_i(t^k) y_{3-i}^k (1 - y_i^k). \tag{B.51}$$

The modulation terms $a_i(t^k)$ and $b_i(t^k)$ are given by

$$a_i(t^k) = \exp\left[R_{3-i}(t^k)\beta\Delta t \left(\frac{\bar{g}_{12}(t^k)}{\bar{g}_i(t^k)} - \bar{g}_{3-i}(t^k) \right) \right], \tag{B.52}$$

$$b_i(t^k) = -\beta \left[\frac{\bar{g}_{12}(t^k)}{\bar{g}_{3-i}(t^k)} - \bar{g}_i(t^k) \right]. \tag{B.53}$$

Appendix C

Appendix to Chapter 3: A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction

Contents

C.1 Derivation of the relationship between the SFA and FLD objective

C.1.1 Derivation for the case of two classes

In this section we derive the expressions for the temporal covariance matrices $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ of the SFA objective (3.2) for the two-class case in terms of the within-class and between-class scatter matrices of the FLD objective (3.4), $\mathbf{S}_W$ and $\mathbf{S}_B$, for the particular method of time series generation described in the main text.

Assume we are given two disjoint point sets $S_1, S_2 \subset \mathbb{R}^n$,

$$S_1 := \{\mathbf{x}_i^1 | i = 1, \ldots, N\}, \tag{C.1}$$

$$S_2 := \{\mathbf{x}_j^2 | j = 1, \ldots, N\}, \tag{C.2}$$

where $\mathbf{x}_i^1$ and $\mathbf{x}_j^2$ denote the data points of class 1 and 2, respectively, and N denotes the number of data points for each of the two classes. Both point sets can be characterized by their mean vectors ($\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$) and covariance matrices ($\boldsymbol{\Sigma}_1$, $\boldsymbol{\Sigma}_2$),

given by

$$\boldsymbol{\mu}_1 = \langle \mathbf{x}_i^1 \rangle = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i^1, \tag{C.3}$$

$$\boldsymbol{\mu}_2 = \langle \mathbf{x}_j^2 \rangle = \frac{1}{N} \sum_{j=1}^{N} \mathbf{x}_j^2, \tag{C.4}$$

and

$$\boldsymbol{\Sigma}_1 = \frac{1}{N} \left\langle \left(\mathbf{x}_i^1 - \boldsymbol{\mu}_1\right) \left(\mathbf{x}_i^1 - \boldsymbol{\mu}_1\right)^T \right\rangle = \sum_{i=1}^{N} \mathbf{x}_i^1 \mathbf{x}_i^{1T} - \boldsymbol{\mu}_1 \boldsymbol{\mu}_1^T, \tag{C.5}$$

$$\boldsymbol{\Sigma}_2 = \frac{1}{N} \left\langle \left(\mathbf{x}_j^2 - \boldsymbol{\mu}_2\right) \left(\mathbf{x}_j^2 - \boldsymbol{\mu}_2\right)^T \right\rangle = \sum_{i=1}^{N} \mathbf{x}_j^2 \mathbf{x}_j^{2T} - \boldsymbol{\mu}_2 \boldsymbol{\mu}_2^T. \tag{C.6}$$

The within-class and between-class scatter matrices of Fisher's linear discriminant are then given by (see (3.6) and (3.5))

$$\begin{aligned} \mathbf{S}_W &= \sum_{i=1}^{N} (\mathbf{x}_i^1 - \boldsymbol{\mu}_1)(\mathbf{x}_i^1 - \boldsymbol{\mu}_1)^T + \sum_{j=1}^{N} (\mathbf{x}_j^2 - \boldsymbol{\mu}_2)(\mathbf{x}_j^2 - \boldsymbol{\mu}_2)^T \\ &= N \left(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2\right) \end{aligned} \tag{C.7}$$

and

$$\mathbf{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \tag{C.8}$$

We now generate a time series $\mathbf{x}_t$ from these two input point sets S_1 and S_2 as described in the main text, using the Markov model in Figure 3.2. We can now express the mean and covariance of this time series $\mathbf{x}_t$ in terms of $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, $\boldsymbol{\Sigma}_1$, and $\boldsymbol{\Sigma}_2$. For the mean we get

$$\boldsymbol{\mu} := \langle \langle \mathbf{x}_t \rangle_t \rangle = \langle \langle \mathbf{x}_t \rangle \rangle_t = \frac{1}{T} \sum_{t=1}^{T} \langle \mathbf{x}_t \rangle = \langle \mathbf{x}_t \rangle = \frac{1}{2} \boldsymbol{\mu}_1 + \frac{1}{2} \boldsymbol{\mu}_2, \tag{C.9}$$

because the stationary distribution of the Markov model in Figure 3.2 is $\boldsymbol{\pi} = \left(\frac{1}{2}, \frac{1}{2}\right)$ (3.10). More generally, the mean of the time series is given by the weighted mean between the two class means, weighted by the probability that a point is drawn from the corresponding class. Note the different expectation operators: $\langle \cdot \rangle_t$ denotes the temporal average over the time series $\mathbf{x}_t$, whereas the average over all possible time series $\mathbf{x}_t$ generated from S_1 and S_2 is given by $\langle \cdot \rangle$. That is, $\langle \langle \mathbf{x}_t \rangle_t \rangle$ refers to the temporal average of a specific time series $\mathbf{x}_t$, averaged over all possible realizations of $\mathbf{x}_t$, whereas $\langle \langle \mathbf{x}_t \rangle \rangle_t$ refers to the temporal average of the expected value of $\mathbf{x}_t$ at a specific time step t. Since this Markov model yields a stationary random process, we can exchange the expectation operators. Similarly, the expected covariance matrix

is given by

$$\begin{aligned}\boldsymbol{\Sigma} := \left\langle \langle (\mathbf{x}_t - \boldsymbol{\mu})(\mathbf{x}_t - \boldsymbol{\mu})^T \rangle_t \right\rangle &= \frac{1}{T}\sum_{t=1}^{T} \langle \mathbf{x}_t \mathbf{x}_t^T \rangle - \boldsymbol{\mu}\boldsymbol{\mu}^T = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{2}(\boldsymbol{\Sigma}_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T) + \frac{1}{2}(\boldsymbol{\Sigma}_2 + \boldsymbol{\mu}_2\boldsymbol{\mu}_2^T) - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{2}\boldsymbol{\Sigma}_1 + \frac{1}{2}\boldsymbol{\Sigma}_2 + \frac{1}{4}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T,\end{aligned} \tag{C.10}$$

where in the last step we used (C.9). Note that the covariance matrix of the time series is not only determined by the covariance matrices of the two classes, but also by their spatial separation as expressed by (C.10). We assume without loss of generality that $\boldsymbol{\mu} = \mathbf{0}$, i.e., $\boldsymbol{\Sigma} = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle$.

Next we consider the covariance matrix of time derivatives. For the expected covariance matrix we write

$$\begin{aligned}\left\langle \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \right\rangle &= \frac{1}{T-1}\sum_{t=2}^{T} \left\langle (\mathbf{x}_t - \mathbf{x}_{t-1})(\mathbf{x}_t - \mathbf{x}_{t-1})^T \right\rangle \\ &= \left(\langle \mathbf{x}_t \mathbf{x}_t^T \rangle + \langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle \right) - \left(\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle + \langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle \right).\end{aligned} \tag{C.11}$$

The two terms in the first part of (C.11) consist of covariances between input samples of the same time index and can be rewritten as (using (C.10))

$$\langle \mathbf{x}_t \mathbf{x}_t^T \rangle \approx \frac{1}{2}\left(\boldsymbol{\Sigma}_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T\right) + \frac{1}{2}\left(\boldsymbol{\Sigma}_2 + \boldsymbol{\mu}_2\boldsymbol{\mu}_2^T\right) \tag{C.12}$$

$$\langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle. \tag{C.13}$$

Because of the stationarity of $\mathbf{x}_t$ the covariance matrix is independent of a time shift. The approximation in (C.12) holds for large T, since the summation in (C.10) contains T terms and the summation in (C.11) contains $T-1$ terms. Similarly, the two terms in the second part of (C.11) consist of the cross-covariances between adjacent time steps. If the classes of $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$ are fixed, then $\mathbf{x}_t$ is chosen independently of $\mathbf{x}_{t-1}$ and we can split up the expectation operator $\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle = \langle \mathbf{x}_{t-1} \rangle \langle \mathbf{x}_t^T \rangle$ into the product of the two class means. Considering the 4 possible class transitions we write

$$\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle = \frac{1}{2}(1-p)\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T + \frac{1}{2}(1-p)\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T + \frac{1}{2}p\boldsymbol{\mu}_1\boldsymbol{\mu}_2^T + \frac{1}{2}p\boldsymbol{\mu}_2\boldsymbol{\mu}_1^T \tag{C.14}$$

$$\langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle^T = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle. \tag{C.15}$$

Plugging (C.12) to (C.15) back into (C.11) yields

$$\left\langle \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \right\rangle = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 + p(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \tag{C.16}$$

In equations (C.10) and (C.16) we have expressed the covariance matrix of the time series $\mathbf{x}_t$, $\langle \mathbf{x}\mathbf{x}^T \rangle_t$, and the covariance matrix of its time derivatives, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$, in

terms of the means and covariances of the two point sets of the FLD problem. We repeat these here for clarity (we drop the expectation $\langle\cdot\rangle$ for convenience):

$$\langle \mathbf{x}\mathbf{x}^T\rangle_t = \frac{1}{2}\boldsymbol{\Sigma}_1 + \frac{1}{2}\boldsymbol{\Sigma}_2 + \frac{1}{4}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T, \tag{C.17}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 + p(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \tag{C.18}$$

Remember that p is the transition probability between the classes, according to Figure 3.2. Recalling the definition of $\mathbf{S}_W$ (C.7) and $\mathbf{S}_B$ (C.8), we finally obtain the result

$$\langle \mathbf{x}\mathbf{x}^T\rangle_t = \frac{1}{2N}\mathbf{S}_W + \frac{1}{4}\mathbf{S}_B, \tag{C.19}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t = \frac{1}{N}\mathbf{S}_W + p \cdot \mathbf{S}_B. \tag{C.20}$$

C.1.2 Derivation for the case of more than two classes

In this section we derive the expressions for the temporal covariance matrices $\langle \mathbf{x}\mathbf{x}^T\rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t$ of the SFA objective (3.2) for the general case of more than two classes in terms of the within-class and between-class scatter matrices of the FLD objective (3.4), $\mathbf{S}_W$ and $\mathbf{S}_B$, for the particular method of time series generation described in the main text. We proceed analogously to the previous section for the two-class case.

Assume we are given C disjoint point sets $S_c \subset \mathbb{R}^n, c = 1, \ldots, C$,

$$S_c := \{\mathbf{x}_i^c | i = 1, \ldots, N_c\}, \tag{C.21}$$

where $\mathbf{x}_i^c$ denote the data points of class c, and N_c denotes the number of data points in each class. Let $N_T = \sum_{c=1}^{C} N_c$ be the total number of points. Each of these point sets can be characterized by its mean vector and covariance matrix, given by

$$\boldsymbol{\mu}_c = \frac{1}{N_c}\sum_{i=1}^{N_c} \mathbf{x}_i^c, \tag{C.22}$$

$$\boldsymbol{\Sigma}_c = \frac{1}{N_c}\sum_{i=1}^{N_c} \mathbf{x}_i^c \mathbf{x}_i^{cT} - \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T. \tag{C.23}$$

The within-class and between-class covariance matrices of the Fisher linear discriminant in the multi-class case are defined by

$$\begin{aligned} \mathbf{S}_W &= \sum_{c=1}^{C}\sum_{i=1}^{N_c} (\mathbf{x}_i^c - \boldsymbol{\mu}_c)(\mathbf{x}_i^c - \boldsymbol{\mu}_c)^T \\ &= \sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c, \end{aligned} \tag{C.24}$$

and

$$\begin{aligned}\mathbf{S}_B &= \sum_{c=1}^{C} N_c(\boldsymbol{\mu}_c - \boldsymbol{\mu})(\boldsymbol{\mu}_c - \boldsymbol{\mu})^T \\ &= \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - N_T \boldsymbol{\mu}\boldsymbol{\mu}^T,\end{aligned} \tag{C.25}$$

where $\boldsymbol{\mu} = 1/N_T \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c$ is the total mean of the input points.

We generate a time series $\mathbf{x}_t$ from these point sets as described in the main text, using the Markov model with states $S = \{1, 2, \ldots, C\}$ and transition probabilities

$$P_{ij} = \begin{cases} a \cdot \frac{N_j}{N_T} & \text{if } i \neq j, \\ 1 - \sum_{k \neq j} P_{ik} & \text{if } i = j, \end{cases} \tag{C.26}$$

for $i, j \in S$. First, we show that

$$\boldsymbol{\pi} = \left(\frac{N_1}{N_T}, \frac{N_2}{N_T}, \ldots, \frac{N_C}{N_T} \right) \tag{C.27}$$

is a stationary distribution of (C.26). This can be easily seen by verifying that for all $j \in S$,

$$\begin{aligned}\pi_j &= \sum_{i \in S} \pi_i P_{ij} \\ &= \sum_{i \neq j} \frac{N_i}{N_T} \cdot a \cdot \frac{N_j}{N_T} + \frac{N_j}{N_T} \left(1 - \sum_{k \neq j} a \cdot \frac{N_k}{N_T} \right) \\ &= a \cdot \frac{N_j}{N_T} \sum_{i \neq j} \frac{N_i}{N_T} + \frac{N_j}{N_T} - a \cdot \frac{N_j}{N_T} \sum_{k \neq j} \frac{N_k}{N_T} \\ &= \frac{N_j}{N_T}. \quad \square\end{aligned} \tag{C.28}$$

For the mean of the time series $\mathbf{x}_t$ we get, analogously to (C.9),

$$\langle \mathbf{x} \rangle_t = \sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c = \frac{1}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c. \tag{C.29}$$

Note that for the particular choice of (C.26) the mean of the time series becomes equal to the total mean of the input points. Similarly, we obtain for the covariance matrix

$$\begin{aligned}\langle \mathbf{x}\mathbf{x}^T \rangle_t &= \sum_{c=1}^{C} \pi_c \left(\boldsymbol{\Sigma}_c + \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T \right) - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c + \frac{1}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{N_T} \mathbf{S}_W + \frac{1}{N_T} \mathbf{S}_B.\end{aligned} \tag{C.30}$$

For the covariance of time derivatives we proceed analogously to the previous section and write

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \left(\langle \mathbf{x}_t \mathbf{x}_t^T \rangle + \langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle \right) - \left(\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle + \langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle \right), \tag{C.31}$$

with

$$\langle \mathbf{x}_t \mathbf{x}_t^T \rangle = \sum_{c=1}^{C} \pi_c \left(\mathbf{\Sigma}_c + \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T \right) \tag{C.32}$$

$$\langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle \tag{C.33}$$

$$\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle = \sum_{i,j \in S} \pi_i P_{ij} \boldsymbol{\mu}_i \boldsymbol{\mu}_j^T \tag{C.34}$$

$$\langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle^T = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle. \tag{C.35}$$

The last equation holds because $\pi_i P_{ij} = \pi_j P_{ji}$. Plugging (C.32) to (C.35) back into (C.31) yields

$$\begin{aligned}
\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t &= 2 \sum_{c=1}^{C} \pi_c \mathbf{\Sigma}_c + 2 \sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - 2 \sum_{i=1}^{C} \sum_{j=1}^{C} \pi_i P_{ij} \boldsymbol{\mu}_i \boldsymbol{\mu}_j^T \\
&= \frac{2}{N_T} \sum_{c=1}^{C} N_c \mathbf{\Sigma}_c + \frac{2}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{2}{N_T} \sum_{c=1}^{C} N_c P_{cc} \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T \\
&\quad - \frac{2}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \left(\sum_{k \neq c} P_{ck} \boldsymbol{\mu}_k^T \right) \\
&= \frac{2}{N_T} \sum_{c=1}^{C} N_c \mathbf{\Sigma}_c + \frac{2}{N_T} \sum_{c=1}^{C} N_c \left(\sum_{k \neq c} a \cdot \frac{N_k}{N_T} \right) \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T \\
&\quad - \frac{2}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \left(\sum_{k \neq c} a \cdot \frac{N_k}{N_T} \boldsymbol{\mu}_k^T \right) \\
&= \frac{2}{N_T} \sum_{c=1}^{C} N_c \mathbf{\Sigma}_c + \frac{2a}{N_T^2} \sum_{c=1}^{C} N_c (N_T - N_c) \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T \\
&\quad - \frac{2a}{N_T^2} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c (N_T \boldsymbol{\mu} - N_c \boldsymbol{\mu}_c)^T \\
&= \frac{2}{N_T} \sum_{c=1}^{C} N_c \mathbf{\Sigma}_c + \frac{2a}{N_T} \left[\sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - N_T \boldsymbol{\mu} \boldsymbol{\mu}^T \right] \\
&= \frac{2}{N_T} \mathbf{S}_W + \frac{2a}{N_T} \mathbf{S}_B.
\end{aligned} \tag{C.36}$$

Finally, we repeat the results (C.30) and (C.36),

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{N_T}\mathbf{S}_W + \frac{1}{N_T}\mathbf{S}_B, \tag{C.37}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{2}{N_T}\mathbf{S}_W + \frac{2a}{N_T}\mathbf{S}_B, \tag{C.38}$$

and note the similarity to the results for the case of two classes in the previous section, (C.19) and (C.20).

C.1.3 Derivation for time series consisting of trajectories

In this section we derive the expressions given in equations (3.21) to (3.24), which reformulate the SFA objective for the case of time series consisting of trajectories of training examples rather than a sequence of individual points that are independently chosen.

First we consider the case where the time series $\mathbf{x}_t$ consists of multiple repetitions of a fixed trajectory $\tilde{\mathbf{t}} := (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_{\tilde{T}})$ of length $\tilde{T}$, and random intervals of independently drawn "noise" samples drawn from the same distribution (characterized by mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$) as the $\tilde{\mathbf{x}}_k$. We assume without loss of generality that $\boldsymbol{\mu} = \langle \mathbf{x}_t \rangle_t = \mathbf{0}$. Furthermore, let T be the total length of $\mathbf{x}_t$, and let $\tilde{p}$ be the fraction of these T time steps of $\mathbf{x}_t$ that are occupied by the trajectory $\tilde{\mathbf{t}}$.

For the expected covariance matrix of $\mathbf{x}_t$ we get

$$\begin{aligned}
\left\langle \langle \mathbf{x}\mathbf{x}^T \rangle_t \right\rangle &= \frac{1}{T}\sum_{t=1}^{T} \langle \mathbf{x}\mathbf{x}^T \rangle \\
&= \frac{1}{T}\sum_{t \in \tilde{\mathbf{t}}} \langle \mathbf{x}\mathbf{x}^T \rangle + \frac{1}{T}\sum_{t \notin \tilde{\mathbf{t}}} \langle \mathbf{x}\mathbf{x}^T \rangle \\
&= \frac{\tilde{p}}{\tilde{T}}\sum_{k=1}^{\tilde{T}} \mathbf{x}_k \mathbf{x}_k^T + (1-\tilde{p})\boldsymbol{\Sigma} \\
&= \tilde{p}\tilde{\boldsymbol{\Sigma}} + (1-\tilde{p})\boldsymbol{\Sigma}.
\end{aligned} \tag{C.39}$$

We use the notation $t \in \tilde{\mathbf{t}}$ to denote that a time step t within the time series $\mathbf{x}_t$ belongs to an instance of $\tilde{\mathbf{t}}$. The matrix

$$\tilde{\boldsymbol{\Sigma}} := \frac{1}{\tilde{T}}\sum_{k=1}^{\tilde{T}} \tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k^T \tag{C.40}$$

is the covariance matrix of $\tilde{\mathbf{t}}$ with itself. Note that the average $\langle \cdot \rangle$ in (C.39) is over all realizations of $\mathbf{x}_t$ with a fixed trajectory $\tilde{\mathbf{t}}$. If we also average over different realizations of $\tilde{\mathbf{t}}$, the covariance becomes $\boldsymbol{\Sigma}$.

The covariance matrix of time derivatives can be written as

$$\begin{aligned}
\left\langle \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \right\rangle &= 2\left\langle \langle \mathbf{x}\mathbf{x}^T \rangle_t \right\rangle - \frac{1}{T-1}\sum_{t=2}^{T} \left(\langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle + \langle \mathbf{x}_{t-1}\mathbf{x}_t^T \rangle \right) \\
&\approx 2 \cdot \tilde{p}\tilde{\boldsymbol{\Sigma}} + 2 \cdot (1-\tilde{p})\boldsymbol{\Sigma} - \tilde{p} \cdot \frac{\tilde{T}-1}{\tilde{T}} \cdot \tilde{\boldsymbol{\Sigma}}_t,
\end{aligned} \tag{C.41}$$

where

$$\tilde{\mathbf{\Sigma}}_t := \frac{1}{\tilde{T}-1}\sum_{k=2}^{\tilde{T}}\left(\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_{k-1}^T + \tilde{\mathbf{x}}_{k-1}\tilde{\mathbf{x}}_k^T\right) \tag{C.42}$$

is the covariance of $\tilde{\mathbf{t}}$ with $\tilde{\mathbf{t}}$ delayed by one time step, i.e., it captures the temporal correlations of time lag 1. This matrix enters equation (C.41) with a coefficient $\tilde{p}(\tilde{T}-1)/\tilde{T}$, because each of the $\tilde{p}T/\tilde{T}$ trajectories of the time series contributes $\tilde{T}-1$ times the expected value (C.42) to the sum in the first line of (C.41). Note that all other temporal correlations of $\mathbf{x}_t$ apart from those caused by $\tilde{\mathbf{t}}$ are zero. The approximation in (C.41) is valid for large T (i.e., when $T/(T-1) \approx 1$).

Inserting (C.39) and (C.41) into the SFA objective (3.2) yields

$$J_{SFA}(\mathbf{w}) = \frac{\mathbf{w}^T\langle\dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t\mathbf{w}}{\mathbf{w}^T\langle\mathbf{x}\mathbf{x}^T\rangle_t\mathbf{w}} = 2 - \tilde{p}\cdot\frac{\tilde{T}-1}{\tilde{T}}\cdot\frac{\mathbf{w}^T\tilde{\mathbf{\Sigma}}_t\mathbf{w}}{\mathbf{w}^T\langle\mathbf{x}\mathbf{x}^T\rangle_t\mathbf{w}}, \tag{C.43}$$

and therefore

$$\min \quad J_{SFA}(\mathbf{w}) \Leftrightarrow \max \quad \frac{\mathbf{w}^T\tilde{\mathbf{\Sigma}}_t\mathbf{w}}{\mathbf{w}^T\langle\mathbf{x}\mathbf{x}^T\rangle_t\mathbf{w}}. \tag{C.44}$$

Next, we consider the two-class problem, where the time series $\mathbf{x}_t$ consists of a sequence of trajectories chosen from two classes $\mathcal{T}_1$ and $\mathcal{T}_2$. After each trajectory, the class of the next trajectory is switched with probability p, or left unchanged with probability $1-p$, according to the Markov model in Figure 3.2. These two trajectory sets can be characterized by their means, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, and their covariances, $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$. Each of these quantities equals equations (C.3) to (C.6) evaluated for point sets S_1 and S_2 composed of the individual points of the trajectories in $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively. Furthermore, let $\tilde{\mathbf{\Sigma}}_t^{(1)}$ and $\tilde{\mathbf{\Sigma}}_t^{(2)}$ be the average temporal covariance matrices with time lag 1 for trajectories in $\mathcal{T}_1$ and $\mathcal{T}_2$, i.e.,

$$\tilde{\mathbf{\Sigma}}_t^{(c)} := \frac{1}{N(\tilde{T}-1)}\sum_{i=1}^{N}\sum_{k=2}^{\tilde{T}}\left[(\tilde{\mathbf{x}}_{i,k}^c - \boldsymbol{\mu}_c)(\tilde{\mathbf{x}}_{i,k-1}^c - \boldsymbol{\mu}_c)^T + (\tilde{\mathbf{x}}_{i,k-1}^c - \boldsymbol{\mu}_c)(\tilde{\mathbf{x}}_{i,k}^c - \boldsymbol{\mu}_c)^T\right] \tag{C.45}$$

where N is the number of trajectories in each of the sets $\mathcal{T}_1$ and $\mathcal{T}_2$ and $\tilde{T}$ is the length of a trajectory (we assume for simplicity that all trajectories have the same length). $\tilde{\mathbf{x}}_{i,k}^c$ is the k-th point in the i-th trajectory of class c. Note that in contrast to (C.42) the mean $\boldsymbol{\mu}_c$ is class-specific and different from zero.

The expected covariance matrix of the time series $\mathbf{x}_t$ is not affected by temporal correlations and is therefore equal to the case where individual points are chosen instead of trajectories (see equation (C.10)):

$$\left\langle\langle\mathbf{x}\mathbf{x}^T\rangle_t\right\rangle = \sum_{c=1}^{C}\pi_c\left(\mathbf{\Sigma}_c + \boldsymbol{\mu}_c\boldsymbol{\mu}_c^T\right), \tag{C.46}$$

where $C = 2$ is the number of classes and $\pi_c = 1/2$ is the probability of being in state c for the stationary distribution of the Markov model. For the expected covariance matrix of time derivatives we write

$$\left\langle\langle\dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t\right\rangle = 2\left\langle\langle\mathbf{x}\mathbf{x}^T\rangle_t\right\rangle - \frac{1}{T-1}\sum_{t=2}^{T}\left(\langle\mathbf{x}_t\mathbf{x}_{t-1}^T\rangle + \langle\mathbf{x}_{t-1}\mathbf{x}_t^T\rangle\right). \tag{C.47}$$

The time series has length T and consists of $T/\tilde{T}$ trajectories. Therefore we can split up the sum in the second term on the right hand side of the last equation into $T - T/\tilde{T}$ contributions from transitions $(\mathbf{x}_{t-1}, \mathbf{x}_t)$ within a trajectory and $T/\tilde{T} - 1$ contributions of switches between two temporally adjacent trajectories (i.e., at time points t when a new trajectory starts). Concerning the first part of the sum, each of the $T/\tilde{T}$ trajectories contributes $\tilde{T} - 1$ times the expected value $\sum_c \pi_c \left(\tilde{\mathbf{\Sigma}}_t^{(c)} + 2\boldsymbol{\mu}_c \boldsymbol{\mu}_c^T \right)$. The second part is determined according to the transition probabilities between the classes, similar to (C.14) and (C.34).

$$\left\langle \langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \right\rangle \approx 2 \sum_{c=1}^{C} \pi_c \mathbf{\Sigma}_c + 2 \sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{\tilde{T} - 1}{\tilde{T}} \sum_{c=1}^{C} \pi_c \tilde{\mathbf{\Sigma}}_t^{(c)} - 2 \left(1 - \frac{1}{\tilde{T}} \right) \sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{2}{\tilde{T}} \sum_{c_1, c_2} \pi_{c_1} P_{c_1 c_2} \boldsymbol{\mu}_{c_1} \boldsymbol{\mu}_{c_2}^T. \quad \text{(C.48)}$$

Again we approximated $T/(T-1) \approx 1$. For the Markov model in Figure 3.2 and $\tilde{\mathbf{\Sigma}}_t = \sum_{c=1}^{C} \pi_c \tilde{\mathbf{\Sigma}}_t^{(c)}$ we can write

$$\left\langle \langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \right\rangle = \mathbf{\Sigma}_1 + \mathbf{\Sigma}_2 - \frac{\tilde{T} - 1}{\tilde{T}} \tilde{\mathbf{\Sigma}}_t + \frac{p}{\tilde{T}} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \quad \text{(C.49)}$$

Using the definitions of the within-class and between-class scatter matrices of the FLD (equations (C.7) and (C.8)) we can rewrite equations (C.46) and (C.49):

$$\langle \mathbf{x} \mathbf{x}^T \rangle_t = \frac{1}{2N\tilde{T}} \mathbf{S}_W + \frac{1}{4} \mathbf{S}_B, \quad \text{(C.50)}$$

$$\langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t = \frac{1}{N\tilde{T}} \mathbf{S}_W + \frac{p}{\tilde{T}} \cdot \mathbf{S}_B - \frac{\tilde{T} - 1}{\tilde{T}} \cdot \tilde{\mathbf{\Sigma}}_t. \quad \text{(C.51)}$$

C.2 Simulation Details

C.2.1 Estimating the error between SFA and FLD

We estimated the deviation between the result of FLD applied to a two-dimensional two-class classification problem and the result of SFA applied to a time series generated from this classification problem using the Markov model in Figure 3.2 as the angle α between the weight vectors yielded by both methods,

$$\alpha = \arccos \frac{\mathbf{w}_{SFA} \cdot \mathbf{w}_{FLD}}{||\mathbf{w}_{SFA}|| \cdot ||\mathbf{w}_{FLD}||}. \quad \text{(C.52)}$$

We evaluated this angular error as a function of p, the switching probability in Figure 3.2, i.e., the probability that two consecutive points in the time series are from different classes.

For each probability p (we varied p from 0.01 to 1.0 linearly in intervals of 0.01) we generated 100 different random classification problems in the following way. For each of the two classes a two-dimensional mean vector and 2-by-2 covariance

matrix was chosen. The coordinates of the mean were drawn independently and uniformly from the interval $[-4, 4]$. The covariance matrix was determined by its two eigenvalues (drawn uniformly from $[0, 1]$) and a rotation angle (drawn uniformly from $[0, 2\pi]$). For each class, 250 points were drawn from a Gaussian distribution with the selected mean and covariance. The time series for SFA is generated using the Markov model in Figure 3.2 with the given switching probability p. The length T of this time series is chosen to be 10000 samples.

We computed the average angle between the weight vectors found by SFA and FLD on those 100 classification problems, yielding values between 0° and 180°. We replaced angles $\alpha > 135°$ with angles $180° - \alpha$, since projection directions with different signs are equivalent. Angles between 45° and 135° were only obtained for $p > 0.5$ where they averaged to about 90°.

C.2.2 Calculating the probability of linear separability

To calculate the probability of linear separability in Figure 3.6B we proceeded in the following way: We generated pairs of point sets (i.e., trajectories) each consisting of 100 points drawn uniformly from the d-dimensional hypercube $[0, 1]^d$. We tested whether these two random point sets are linearly separable using an efficient method proposed in (Yogananda et al., 2007). We evaluated the probability of linear separability for each dimension d as the percentage of 1000 such randomly generated classification problems that resulted in linearly separable point sets. For each classification problem we also searched for the minimal distance between any two points from different sets. We calculated the average minimum distance over all 1000 classification problems for each dimension d.

We found that the curve for the probability of linear separability closely resembles the analytical result of (Cover, 1965), which considered the fraction of all possible dichotomies of N given data points in general position in d dimensions which are linearly separable.

C.2.3 Detailed description of the network simulations

C.2.3.1 Generation of input spike trains

In our circuit simulations we use two different types of input: spike trains generated from isolated spoken digits preprocessed with a model of the cochlea and spike patterns embedded in a continuous stream of Poisson input.

In the speech recognition tasks we use the isolated spoken digits dataset in (Hopfield and Brody, 2000, 2001). This dataset consists of of the audio signals recorded from 5 speakers pronouncing the digits "zero", "one", ..., "nine" in ten different utterances (trials) each, i.e., overall there are 500 speech samples. The duration of an utterance is several 100ms.

To generate a biologically realistic network input, the raw audio signals are converted into the output of a cochlea ("cochleagram") using Lyon's Passive Ear model (Lyon, 1982). This computational model consists of a linear filterbank and a nonlinear gain control network and captures the filtering properties of the cochlea and hair cells of the inner ear. The resulting analog cochleagram is a 86-dimensional

time series with values between 0 and 1. An implementation of this cochlea model can be found in the Auditory Toolbox for Matlab (Slaney, 1998).

This analog waveform is then transformed into spike trains using the BSA algorithm (Schrauwen and Campenhout, 2003). This method is able to reconstruct a spike train from an analog trace with a given reconstruction filter. Filtering the spike train with this reconstruction filter should yield a trace with a minimal deviation from the original waveform. We used the implementation from the Reservoir Computing Toolbox (Verstraeten et al., 2007). We chose a reconstruction filter with an exponential form ($\tau = 30$ms) and selected the threshold parameter of the algorithm to be 0.97 (standard value). In order to obtain lower firing rates of the spiking stimuli, we scaled the amplitude of the reconstruction filter such that it has an integral of 40. Furthermore we selected 20 from these 86 spike trains in equidistant steps. The same spike patterns have also been used for the speech recognition task in (Legenstein et al., 2008).

The embedded spike patterns, on the other hand, consist of 10 Poisson spike train segments of length $T_{seg} = 250$ms and with rate $r = 20$Hz. Poisson spike trains are generated by positioning spikes in time according to inter-spike intervals drawn from an exponential distribution with rate r until the segment length T_{seg} is reached. Additionally a refractory period of 3ms after a spike is considered, during which no further spike can occur. Similar spike patterns have been considered for example in (Häusler and Maass, 2007). For each pattern class one such pattern is generated. To model the continuous Poisson input, we preceded each pattern instance with a random Poisson input with a duration uniformly drawn between 100ms and 500ms.

C.2.3.2 Our model of a cortical microcircuit

As a cortical microcircuit we use the laminar circuit model from (Häusler and Maass, 2007) consisting of 560 spiking neurons (Izhikevich neuron model) with dynamic conductance-based synapses. The short-term dynamics of these synapses has been modelled according to the phenomenological model proposed in (Markram et al., 1998). To reproduce the background synaptic input that cortical neurons typically receive *in vivo*, additional synaptic noise is incorporated as an Ornstein-Uhlenbeck (OU) process as conductance input (Destexhe et al., 2001). All parameters of this model, including short-term synaptic dynamics and background synaptic activity, are chosen as in (Häusler and Maass, 2007).

The neurons are organized in six pools; an excitatory and inhibitory pool for each of the layers 2/3, 4, and 5. The numbers of neurons in each layer are 168, 112, and 280, respectively. The connection strengths and probabilities within a pool and between the pools are obtained from data found in (Thomson et al., 2002) and (Gupta et al., 2000). All of the stimulus spike trains (5 for the spike pattern task; 20 for the speech recognition task) are fed into the circuit via the input stream that connects mainly to Layer 4 ("input stream 1" in Figure 1 of (Häusler and Maass, 2007)). The second input stream into Layer 2/3 is switched off.

C.2.3.3 Training the readouts of the circuit

We instantiated a single circuit and simulated the same network for each stimulus in all the experiments described in this article. In the speech recognition tasks the network is simulated for the same amount of time for all stimuli (500ms for Figure 3.8 and 750ms for Figure 3.9). We low-pass filtered the response spike trains with an exponential filter in order to model the contribution of these spikes to the membrane potential of a hypothetical readout neuron. The time constant of this exponential filter is chosen to be 30ms. We refer to this low-pass filtered high-dimensional analog trace as the trajectory of network states in response to a particular stimulus.

To generate a training input for SFA we sampled these trajectories with a sampling time of 1ms and concatenated a random sequence of such trajectories in time (100 trajectories for Figure 3.8; 1000 trajectories for Figure 3.9; 200 trajectories for Figure 3.7). For the embedded spike pattern task one trajectory is defined by the response during one noise/pattern pair. Note that the same stimulus yields different trajectories due to the intrinsic OU-noise of the network that is used to model the background synaptic activity. We proceeded in a similar way as we generated the time series from a classification problem: After each drawing of a trajectory we switched the class from which the next trajectory is drawn according to a Markov model such as that in Figure 3.2. The probability p for switching the class is chosen to be 0.2 for all experiments, except for the experiment in Figure 3.7 we had to choose a lower value of $p = 0.01$. We ensured that in the resulting training sequence the number of trajectories was balanced across different classes by requiring that the standard deviation of the numbers of trajectories for each class was at most $T/20$. Before applying SFA or FLD, we projected the trajectories onto the first 100 principal components in order to prevent the covariance matrices from becoming singular, which would lead to numerical issues in the corresponding eigenvalue problems. For the SVM classification of the network states in Figure 3.8A we used a linear kernel with $C = 10$. The training set for both FLD and SVM consisted of the network states sampled every 1ms of all trajectories considered, but only states during stimulus presentation are taken into account. The same applies to the SVM classification of the slow features for the evaluation of the SFA performance. This performance is evaluated using 10-fold stratified cross validation, where the folds are sampled according to the class size.

C.2.4 Software

We performed all simulations using Python and NumPy. We used the implementations of SFA and FLD contained in the MDP toolkit (Zito et al., 2008). The Modular toolkit for Data Processing (MDP) is a data processing framework written in Python. The circuit simulations were carried out with the PCSIM software package (http://www.lsm.tugraz.at/pcsim). PCSIM is a parallel simulator for biologically realistic neural networks with a fast C++ simulation core and a Python interface. For Support Vector Machines (SVM) we used the libSVM toolbox contained in the PyML package (http://pyml.sourceforge.net/). Figures were created using Python/Matplotlib and Matlab.

Bibliography

Anderson, L. A., Christianson, G. B., and Linden, J. F. (2009). Stimulus-Specific Adaptation Occurs in the Auditory Thalamus. *J Neurosci*, 29(22):7359–7363. 101

Arabzadeh, E., Panzeri, S., and Diamond, M. E. (2004). Whisker vibration information carried by rat barrel cortex neurons. *J Neurosci*, 24(26):6011–20. 79

Arabzadeh, E., Panzeri, S., and Diamond, M. E. (2006). Deciphering the spike train of a sensory neuron: counts and temporal patterns in the rat whisker pathway. *J Neurosci*, 26(36):9216–9226. 79

Asari, H. and Zador, A. M. (2009). Long-lasting context dependence constrains neural encoding models in rodent auditory cortex. *J Neurophysiol*, 102(5):2638–2656. 99

Atick, J. J. and Redlich, A. N. (1993). Convergent algorithm for sensory receptive field development. *Neural Computation*, 5:45–60. 42

Bartlett, E. L. and Wang, X. (2005). Long-lasting modulation by stimulus context in primate auditory cortex. *J Neurophysiol*, 94(1):83–104. 78, 99

Becker, S. (1996). Mutual information maximization: Models of cortical self-organization. *Network: Computation in Neural Systems*, 7:7–31. 35

Becker, S. and Hinton, G. E. (1992). A self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355(6356):161–163. 40, 68

Bell, A. J. and Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159. 35

Ben-Hur, A., Ong, C. S., Sonnenburg, S., Schölkopf, B., and Rätsch, G. (2008). Support vector machines and kernels for computational biology. *PLoS Comput Biol*, 4(10):e1000173. 83

Berkes, P. (2005a). Handwritten digit recognition with nonlinear fisher discriminant analysis. In *Proc. of ICANN Vol. 2*, volume 3696 of *Lecture Notes in Computer Science*, pages 285–287. Springer. 73

Berkes, P. (2005b). Pattern recognition with slow feature analysis. Cognitive Sciences EPrint Archive (CogPrint) 4104. `http://cogprints.org/4104/`. 72, 73

Berkes, P. (2006). *Temporal slowness as an unsupervised learning principle*. PhD thesis, Humboldt-Universität zu Berlin. 52, 72, 73

Berkes, P. and Wiskott, L. (2003). Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 5(6):579–602. 44, 73

Berkes, P. and Wiskott, L. (2006). On the analysis and interpretation of inhomogeneous quadratic forms as receptive fields. *Neural Computation*, 18(8):1868–1895. 73

Bernacchia, A., Seo, H., Lee, D., and Wang, X.-J. (2011). A reservoir of time constants for memory traces in cortical neurons. *Nature Neuroscience*, 14(3):366–372. 78

Bienenstock, E. L., Cooper, L. N., and Munro, P. W. (1982). Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *J Neurosci*, 2(1):32–48. 7, 15, 19, 20, 35

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA. 45, 83

Blaschke, T., Berkes, P., and Wiskott, L. (2006). What is the relation between slow feature analysis and independent component analysis? *Neural Computation*, 18(10):2495–2508. 52, 74

Blaschke, T., Zito, T., and Wiskott, L. (2007). Independent slow feature analysis and nonlinear blind source separation. *Neural Computation*, 19(4):994–1021. 52, 74

Bray, A. and Martinez, D. (2003). Kernel-based extraction of slow features: Complex cells learn disparity and translation invariance from natural images. In *Proc. of NIPS 2002, Advances in Neural Information Processing Systems*. MIT Press. 70

Broome, B. M., Jayaraman, V., and Laurent, G. (2006). Encoding and decoding of overlapping odor sequences. *Neuron*, 51:467–482. 43

Brosch, M. and Schreiner, C. E. (2000). Sequence sensitivity of neurons in cat primary auditory cortex. *Cerebral cortex (New York, N.Y. : 1991)*, 10(12):1155–1167. 78, 80, 85, 99

Brosch, M., Schulz, A., and Scheich, H. (1999). Processing of sound sequences in macaque auditory cortex: response enhancement. *J Neurophysiol*, 82(3):1542–59. 78, 80, 99

Buonomano, D. and Maass, W. (2009). State-dependent computations: Spatiotemporal processing in cortical networks. *Nature Reviews in Neuroscience*, 10(2):113–125. 43, 71, 75, 78, 79, 101

Buonomano, D. V. and Mauk, M. D. (1994). Neural network model of the cerebellum: temporal discrimination and the timing of motor responses. *Neural Computation*, 6:38–55. 71

Calvert, G., Spence, C., and Stein, B. (2004). *The Handbook of Multisensory Processes*. MIT Press, Cambridge. 37

Chang, C.-C. and Lin, C.-J. (2001). *LIBSVM: a library for support vector machines*. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. 84

Condon, C. D. and Weinberger, N. M. (1991). Habituation produces frequency-specific plasticity of receptive fields in the auditory cortex. *Behavioral Neuroscience*, 105(3):416–430. 102

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, 14(3):326–334. 57, 126

Cover, T. M. and Thomas, J. A. (1991). *Elements of Information Theory*. Wiley, New York. 9, 10, 11, 22, 78, 80, 82

Cox, D. D., Meier, P., Oertelt, N., and DiCarlo, J. J. (2005). "Breaking" position-invariant object recognition. *Nature Neuroscience*, 8(9):1145–1147. 40

Creutzig, F. and Sprekeler, H. (2008). Predictive coding and the slowness principle: an information-theoretic approach. *Neural Computation*, 20(4):1026–1041. 74

David, S. V., Mesgarani, N., Fritz, J. B., and Shamma, S. A. (2009). Rapid synaptic depression explains nonlinear modulation of spectro-temporal tuning in primary auditory cortex by natural stimuli. *J Neurosci*, 29(11):3374–86. 79, 102

Dayan, P. and Abbott, L. F. (2001). *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. MIT-Press. 42

Destexhe, A., Rudolph, M., Fellous, J. M., and Sejnowski, T. J. (2001). Fluctuating synaptic conductances recreate in vivo-like activity in neocortical neurons. *Neuroscience*, 107(1):13–24. 70, 127

DeWeese, M. R., Hromadka, T., and Zador, A. M. (2005). Reliability and representational bandwidth in the auditory cortex. *Neuron*, 48(3):479–88. 99

DeWeese, M. R., Wehr, M., and Zador, A. M. (2003). Binary spiking in auditory cortex. *J Neurosci*, 23(21):7940–9. 99

DiCarlo, J. J. and Cox, D. D. (2007). Untangling invariant object recognition. *Trends in Cognitive Sciences*, 11(8):333–341. 40

Doupe, A. J. (1997). Song- and order-selective neurons in the songbird anterior forebrain and their emergence during vocal development. *J Neurosci*, 17(3):1147–67. 78, 99

Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification*. Wiley. 45, 49, 50

Einhäuser, W., Kayser, C., König, P., and Körding, K. P. (2002). Learning the invariance properties of complex cells from their responses to natural stimuli. *European Journal of Neuroscience*, 15(3):475–486. 74

Elhilali, M., Fritz, J. B., Klein, D. J., Simon, J. Z., and Shamma, S. A. (2004). Dynamics of precise spike timing in primary auditory cortex. *J Neurosci*, 24(5):1159–1172. 102

Euston, D. R., Tatsuno, M., and McNaughton, B. L. (2007). Fast-forward playback of recent memory sequences in prefrontal cortex during sleep. *Science*, 318(5853):1147–1150. 43

Fairhall, A. L., Lewen, G. D., Bialek, W., and de Ruyter van Steveninck, R. R. (2001). Efficiency and ambiguity in an adaptive neural code. *Nature*, 412(6849):787–792. 102

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annuals of Eugenics*, 7:179–188. 45

Földiak, P. (1991). Learning invariance from tranformation sequences. *Neural Computation*, 3:194–200. 40, 68, 74

Franzius, M., Sprekeler, H., and Wiskott, L. (2007a). Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS Computational Biology*, 3(8):e166. 74

Franzius, M., Vollgraf, R., and Wiskott, L. (2007b). From grids to places. *Journal of Computational Neuroscience*, 22(3):297–299. 74

Franzius, M., Wilbert, N., and Wiskott, L. (2008). Invariant object recognition with slow feature analysis. In Kurkova, V., Neruda, R., and Koutnik, J., editors, *Proc. 18th Intl. Conf. on Artificial Neural Networks, ICANN'08, Prague*, volume 5163 of *Lecture Notes in Computer Science*, pages 961–970. Springer. 73

Fritz, J., Shamma, S., Elhilali, M., and Klein, D. (2003). Rapid task-related plasticity of spectrotemporal receptive fields in primary auditory cortex. *Nature Neuroscience*, 6(11):1216–1223. 37

Gerstner, W. and Kistler, W. M. (2002). *Spiking Neuron Models*. Cambridge University Press, Cambridge. 9

Goodall, M. C. (1960). Statistics: Performance of a stochastic net. *Nature*, 185(4712):557–558. 42

Gupta, A., Wang, Y., and Markram, H. (2000). Organizing principles for a diversity of GABAergic interneurons and synapses in the neocortex. *Science*, 287:273–278. 43, 59, 127

Gütig, R., Aharonov, R., Rotter, S., and Sompolinsky, H. (2003). Learning input correlations through non-linear temporally asymmetric hebbian plasticity. *J Neurosci*, 23:3697–3714. 27

Häusler, S. and Maass, W. (2007). A statistical analysis of information processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, 17(1):149–162. 43, 59, 61, 63, 68, 70, 127

Hebb, D. O. (1949). *The Organization of Behavior*. Wiley. 1

Hopfield, J. J. and Brody, C. D. (2000). What is a moment? "Cortical" sensory integration over a brief interval. *Proc. Nat. Acad. Sci. USA*, 97(25):13919–13924. 61, 68, 126

Hopfield, J. J. and Brody, C. D. (2001). What is a moment? Transient synchrony as a collective mechanism for spatio-temporal integration. *Proc. Nat. Acad. Sci. USA*, 98(3):1282–1287. 61, 126

Hromádka, T., Deweese, M. R., and Zador, A. M. (2008). Sparse representation of sounds in the unanesthetized auditory cortex. *PLoS biology*, 6(1):e16+. 99

Hromadka, T. and Zador, A. M. (2009). Representations in auditory cortex. *Curr Opin Neurobiol*, 19(4):430–3. 100

Hyvärinen, A., Karhunen, J., and Oja, E. (2001). *Independent Component Analysis*. Wiley, New York. 6, 34, 74

Hyvärinen, A. and Oja, E. (1996). Simple neuron models for independent component analysis. *Int. Journal of Neural Systems*, 7(6):671–687. 34

Hyvärinen, A. and Oja, E. (1998). Independent component analysis by general non-linear Hebbian-like learning rules. *Signal Processing*, 64(3):301–313. 34

Ince, R. A. A., Petersen, R. S., Swan, D. C., and Panzeri, S. (2009). Python for information theoretic analysis of neural data. *Frontiers in Neuroinformatics*, 3(4). 83

Izhikevich, E. M. (2007). Solving the distal reward problem through linkage of STDP and dopamine signaling. *Cerebral Cortex*, 17:2443–2452. 74

Ji, D. and Wilson, M. A. (2008). Firing rate dynamics in the hippocampus induced by trajectory learning. *J Neurosci*, 28(18):4679–4689. 43

Jones, L. M., Fontanini, A., Sadacca, B. F., Miller, P., and Katz, D. B. (2007). Natural stimuli evoke dynamic sequences of states in sensory cortical ensembles. *Proc. Nat. Acad. Sci. USA*, 104(47):18772–18777. 43

Kayser, C., Logothetis, N. K., and Panzeri, S. (2010). Millisecond encoding precision of auditory cortex neurons. *Proc. Nat. Acad. Sci. USA*, 107(39):16976–16981. 100

Kempter, R., Gerstner, W., and van Hemmen, J. L. (1999). Hebbian learning and spiking neurons. *Phys. Rev. E*, 59(4):4498–4514. 13

Kilgard, M. P. and Merzenich, M. M. (1999). Distributed representation of spectral and temporal information in rat primary auditory cortex. *Hearing Research*, 134(1-2):16–28. 78, 99

Legenstein, R. and Maass, W. (2007). Edge of chaos and prediction of computational performance for neural microcircuit models. *Neural Networks*, 20(3):323–334. 72

Legenstein, R., Näger, C., and Maass, W. (2005). What can a neuron learn with spike-timing-dependent plasticity? *Neural Computation*, 17(11):2337–2382. 2, 27, 69

Legenstein, R., Pecevski, D., and Maass, W. (2008). A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS Computational Biology*, 4(10):1–27. 2, 61, 70, 74, 127

Lewicki, M. S. and Arthur, B. J. (1996). Hierarchical organization of auditory temporal context sensitivity. *J Neurosci*, 16(21):6987–98. 78, 99

Li, N. and DiCarlo, J. J. (2008). Unsupervised natural experience rapidly alters invariant object representation in visual cortex. *Science*, 321:1502–1507. 40, 42

Linsker, R. (1989). How to generate ordered maps by maximizing the mutual information between input and output signals. *Neural Computation*, 1:402–411. 35

Lu, T., Liang, L., and Wang, X. (2001). Temporal and rate representations of time-varying signals in the auditory cortex of awake primates. *Nature Neuroscience*, 4(11):1131–1138. 100

Lyon, R. F. (1982). A computational model of filtering, detection, and compression in the cochlea. In *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing*, pages 1282–1285. 61, 126

Maass, W. (2007). Liquid computing. In *Proceedings of the Conference CiE'07: COMPUTABILITY IN EUROPE 2007, Siena (Italy)*, Lecture Notes in Computer Science, pages 507–516. Springer (Berlin). 2

Maass, W. (2010). Liquid state machines: Motivation, theory, and applications. In Cooper, B. and Sorbi, A., editors, *Computability in Context: Computation and Logic in the Real World*. Imperial College Press. in press. 2

Maass, W., Joshi, P., and Sontag, E. D. (2007). Computational aspects of feedback in neural circuits. *PLoS Computational Biology*, 3(1):e165, 1–20. 2

Maass, W. and Markram, H. (2004). On the computational power of recurrent circuits of spiking neurons. *Journal of Computer and System Sciences*, 69(4):593–616. 2

Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560. 2, 58, 61, 69, 78, 101

Maass, W., Natschläger, T., and Markram, H. (2004a). Computational models for generic cortical microcircuits. In Feng, J., editor, *Computational Neuroscience: A Comprehensive Approach*, chapter 18, pages 575–605. Chapman & Hall/CRC, Boca Raton. 2, 61, 67, 69

Maass, W., Natschläger, T., and Markram, H. (2004b). Fading memory and kernel properties of generic cortical microcircuit models. *Journal of Physiology – Paris*, 98(4–6):315–330. 2

Malone, B. J., Scott, B. H., and Semple, M. N. (2002). Context-dependent adaptive coding of interaural phase disparity in the auditory cortex of awake macaques. *J Neurosci*, 22:4625–4638. 78, 99, 102

Malone, B. J. and Semple, M. N. (2001). Effects of auditory stimulus context on the representation of frequency in the gerbil inferior colliculus. *J Neurophysiol*, 86(3):1113–1130. 102

Margoliash, D. and Fortune, E. S. (1992). Temporal and harmonic combination-sensitive neurons in the zebra finch's hvc. *J Neurosci*, 12(11):4309–26. 78, 99

Markram, H., Toledo-Rodriguez, M., Wang, Y., Gupta, A., Silberberg, G., and Wu, C. (2004). Interneurons of the neocortical inhibitory system. *Nature Reviews in Neuroscience*, 5:793–807. 1

Markram, H., Wang, Y., and Tsodyks, M. (1998). Differential signaling via the same axon of neocortical pyramidal neurons. *Proc. Nat. Acad. Sci. USA*, 95:5323–5328. 1, 127

Masquelier, T., Guyonneau, R., and Thorpe, S. J. (2009). Competitive STDP-based spike pattern learning. *Neural Computation*, 21(5):1259–1276. 74

Masquelier, T. and Thorpe, S. J. (2007). Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Computational Biology*, 3(2):e31. 74

Mazor, O. and Laurent, G. (2005). Transient dynamics versus fixed points in odor representations by locust antennal lobe projection neurons. *Neuron*, 48:661–673. 43

McKenna, T. M., Weinberger, N. M., and Diamond, D. M. (1989). Responses of single auditory cortical neurons to tone sequences. *Brain research*, 481(1):142–153. 78, 99

Miller, G. A. (1955). Note on the bias of information estimates. *Information Theory in Psychology; Problems and Methods*, pages 95–100. 79, 81

Mitchison, G. (1991). Removing time variation with the anti-hebbian differential synapse. *Neural Computation*, 3(3):312–320. 40, 41, 68

Montemurro, M. A., Rasch, M. J., Murayama, Y., Logothetis, N. K., and Panzeri, S. (2008). Phase-of-firing coding of natural visual stimuli in primary visual cortex. *Current Biology*, 18(5):375–380. 79

Nikolic, D., Haeusler, S., Singer, W., and Maass, W. (2009). Distributed fading memory for stimulus properties in the primary visual cortex. *PLoS Biology*, 7(12):1–19. 2, 78, 79, 83, 84, 95, 96, 99, 101, 102

Panzeri, S., Senatore, R., Montemurro, M. A., and Petersen, R. S. (2007). Correcting for the sampling bias problem in spike train information measures. *J Neurophysiol*, 98:1064–1072. 79, 81, 82, 99

Panzeri, S. and Treves, A. (1996). Analytical estimates of limited sampling biases in different information measures. *Network: Computation in Neural Systems*, 7:87–107. 79, 81

Rabinovich, M., Huerta, R., and Laurent, G. (2008). Transient dynamics for neural processing. *Science*, 321:48–50. 43

Rieke, F., Warland, D., de Ruyter van Steveninck, R., and Bialek, W. (1997). *Spikes - Exploring the Neural Code*. MIT Press. 81, 100

Sadagopan, S. and Wang, X. (2009). Nonlinear spectrotemporal interactions underlying selectivity for complex sounds in auditory cortex. *J Neurosci*, 29(36):11192–11202. 100

Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels*. MIT Press, Cambridge, MA. 41, 65, 79, 83

Schrauwen, B. and Campenhout, J. V. (2003). BSA, a fast and accurate spike train encoding scheme. In *Proceedings of the International Joint Conference on Neural Networks*. 127

Shannon, C. E. (1948). A mathematical theory of communication. *AT&T Bell Labs. Tech. J.*, 27:379–423. 80

Shuler, M. G. and Bear, M. F. (2006). Reward timing in the primary visual cortex. *Science*, 311(5767):1606–1609. 37

Sigala, N. and Logothetis, N. K. (2002). Visual categorization shapes feature selectivity in primate temporal cortex. *Nature*, 415:318–320. 37

Singh, N. C. and Theunissen, F. E. (2003). Modulation spectra of natural sounds and ethological theories of auditory processing. *The Journal of the Acoustical Society of America*, 114(6):3394–3411. 99

Sipser, M. (1996). *Introduction to the Theory of Computation*. International Thomson Publishing, 1st edition. 1

Slaney, M. (1998). Auditory Toolbox: A MATLAB toolbox for auditary modeling work. Technical report, Apple Computer, Inc. Apple Computer Technical Report #45. 127

Slonim, N. (2002). *The Information Bottleneck: Theory and Applications*. PhD thesis, Hebrew University, Jerusalem. 22

Sprekeler, H., Michaelis, C., and Wiskott, L. (2007). Slowness: An objective for spike-timing-plasticity? *PLoS Computational Biology*, 3(6):1136–1148. 41, 74

Stone, J. V. and Bray, A. J. (1995). A learning rule for extracting spatio-temporal invariances. *Network: Computation in Neural Systems*, 6(3):429–436. 40, 68

Sussillo, D. and Abbott, L. D. (2009). Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63:544–557. 78

Thomson, A. M., West, D. C., Wang, Y., and Bannister, A. P. (2002). Synaptic connections and small circuits involving excitatory and inhibitory neurons in layers 2–5 of adult rat and cat neocortex: triple intracellular recordings and biocytin labelling in vitro. *Cerebral Cortex*, 12(9):936–953. 43, 59, 127

Tishby, N., Pereira, F. C., and Bialek, W. (1999). The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377. 6, 20, 21, 22, 23, 74

Toyoizumi, T., Pfister, J.-P., Aihara, K., and Gerstner, W. (2005). Generalized Bienenstock-Cooper-Munro rule for spiking neurons that maximizes information transmission. *Proc. Natl. Acad. Sci. USA*, 102:5239–5244. 6, 7, 8, 9, 10, 11, 12, 13, 15, 16, 18, 20, 22, 35, 106, 108, 109

Turner, R. and Sahani, M. (2007). A maximum-likelihood interpretation for slow feature analysis. *Neural Computation*, 19(4):1022–1038. 75

Ulanovsky, N., Las, L., Farkas, D., and Nelken, I. (2004). Multiple time scales of adaptation in auditory cortex neurons. *J Neurosci*, 24(46):10440–10453. 80, 85, 99, 102

Ulanovsky, N., Las, L., and Nelken, I. (2003). Processing of low-probability sounds by cortical neurons. *Nature Neuroscience*, 6(4):391–398. 80, 102

Verstraeten, D., Schrauwen, B., D'Haene, M., and Stroobandt, D. (2007). An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391 – 403. Echo State Networks and Liquid State Machines. 127

Verstraeten, D., Schrauwen, B., Stroobandt, D., and Campenhout, J. V. (2005). Isolated word recognition with the liquid state machine: a case study. *Inf. Process. Lett.*, 95(6):521–528. 61, 68

Vinje, W. E. and Gallant, J. L. (2000). Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, pages 1273–1275. 42

Wallis, G. and Rolls, E. T. (1997). A model of invariant object recognition in the visual system. *Prog. Neurobiol*, 51:167–194. 74

Wang, X., Lu, T., Snider, R. K., and Liang, L. (2005). Sustained firing in auditory cortex evoked by preferred stimuli. *Nature*, 435(7040):341–6. 100

Wehr, M. and Zador, A. M. (2005). Synaptic mechanisms of forward suppression in rat auditory cortex. *Neuron*, 47:437–445. 102

Wiskott, L. (1998). Learning invariance manifolds. In *Proc. of the 5th Joint Symp. on Neural Computation, May 16, San Diego, CA*, volume 8, pages 196–203, San Diego, CA. Univ. of California. 40, 41

Wiskott, L. (2003). Slow feature analysis: A theoretical analysis of optimal free responses. *Neural Computation*, 15(9):2147–2177. 41, 53, 56, 64, 69, 71

Wiskott, L. and Sejnowski, T. J. (2002). Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14(4):715–770. 40, 41, 44, 56, 63, 64, 67, 68, 69, 70, 71

Wyss, R., König, P., and Verschure, P. F. M. J. (2006). A model of the ventral visual system based on temporal stability and local memory. *PLoS Computational Biology*, 4(5):0001–0008. 74

Yin, P., Fritz, J. B., and Shamma, S. A. (2010). Do ferrets perceive relative pitch? *The Journal of the Acoustical Society of America*, 127(3):1673–1680. 101

Yin, P., Mishkin, M., Sutter, M., and Fritz, J. B. (2008). Early stages of melody processing: stimulus-sequence and task-dependent neuronal activity in monkey auditory cortical fields A1 and R. *J Neurophysiol*, 100(6):3009–3029. 78, 99

Yogananda, A. P., Murthy, M. N., and Gopal, L. (2007). A fast linear separability test by projection of positive points on subspaces. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 713–720, New York, NY, USA. ACM. 126

Zito, T., Wilbert, N., Wiskott, L., and Berkes, P. (2008). Modular toolkit for Data Processing (MDP): a Python data processing framework. *Frontiers in Neuroinformatics*, 2. 128

Zucker, R. S. and Regehr, W. G. (2002). Short-term synaptic plasticity. *Annual Review of Physiology*, 64:355–405. 102