Graz, University of Technology

Institute for Theoretical Computer Science (IGI),

Graz University of Technology

A-8010 Graz, Austria

Master's Thesis

Structure and Self Organization in Spiking Neural Networks

Submitted by:

Gernot Griesbacher

Supervisor:

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

Graz University of Technology, Austria

December, 2013

Technische Universität Graz

Institut für Grundlagen der Informationsverarbeitung (IGI),

Technische Universität Graz

A-8010 Graz

Masterarbeit

Struktur und Selbstorganisation in Spikenden Neuronalen Netzen

Vorgelegt von:

Gernot Griesbacher

Betreuer:

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

Technische Universität Graz, Österreich

Dezember 2013

Abstract

Sequentially activated ensembles of neurons, called assemblies, play a key role in neural computations. However, the principles underlying the emergence are still in question.

In this thesis biological inspired neural network simulations were carried out to analyze the emergence and self organization of assemblies in different connectivity structures. The network consists of recurrently connected stochastic spiking Sparse Winner-Take-All (SWTA) microcircuit models that learn through Spike-Timing-Dependent Plasticity (STDP).

We provide here principles and results, how to construct networks that are capable to emerge clear input-specific assembly activation trajectories, even in presence of high amount of noise and very little external pattern stimulation. Specific assemblies emerge self organized for various different stimuli and they are also spatially distributed. Furthermore the same neurons could be part of different assembly groups. It is also possible to scale up the presented structures.

These findings suggest that the presented networks could mimic various salient biological features, which probably could contribute for example to the modeling of larger neural structures or generating neural assembly codes.

Kurzfassung

Sequentiell aktivierte Neuronengruppen, auch Assemblies genannt, haben eine wichtige Bedeutung für neuronale Prozesse. Jedoch sind die Prinzipien wie diese entstehen noch nicht gänzlich geklärt.

Es wurden biologisch inspirierte neuronale Netzwerksimulationen durchgeführt, um Auftreten und Selbstorganisation von Assemblies in verschiedenen Strukturen zu untersuchen. Die Netzwerke bestehen aus rekurrent verbundenen stochastischen Sparse Winner-Take-All (SWTA) Netzwerkmodellen, die mithilfe Spike-Timing-Dependent Plasticity (STDP) lernen.

Diese Arbeit liefert Prinzipien und Ergebnisse, wie Netzwerke erstellt werden können, die fähig sind klare eingangsspezifische Aktivierungstrajektorien von Assemblies entstehen zu lassen. Dies ist auch möglich, wenn ein hoher Rauschanteil bei sehr wenig externer Stimulation dem Netzwerk zugeführt wird. Spezifische Assemblies entstehen selbst organisiert für verschiedene Stimuli. Neuronen einer Assembly sind räumlich verteilt und können auch Teil mehrerer Assemblies sein. Die präsentierten Strukturen sind zudem skalierbar.

Die Ergebnisse weisen darauf hin, dass diese Netzwerke biologische Eigenschaften replizieren können und daher zum Beispiel zur Modellierung von größeren neuronalen Strukturen beitragen oder das Generieren von neuronalen Assemblycodes genutzt werden können.

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.

....	....
(Place, Date)	(Signature)

Eidesstattliche Erklärung

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

....	....
(Ort, Datum)	(Unterschrift)

Contents

List of Figures

Acknowledgements

I want to thank Zeno Jonke, Prof. Maass and all IGI colleagues for the help, the countless ideas and valuable discussions that made this thesis possible.

I owe my deepest gratitude to my parents, Herbert and Maria for their unconditional support and patience. Especially I am thankful to Mona and my little son Adrian. Thank you for the countless positive distractions and lessons to remember me what is really important in life.

Gernot Griesbacher
Graz, Austria, Dezember, 2013

Chapter 1

Introduction

'Cell assemblies' are considered as basic building block for neural activity. According to [Hebb, 1949] strongly connected groups of neurons represents a cognitive entity, that could be chained by internal processes called 'phase sequences' to form a basis for complex cognitive tasks.
But what is the mechanism that let neurons organize themselves to form this structured spiking behavior?

[Klampfl and Maass, 2013] presented networks that automatically emerge assemblies through Spike-Timing-Dependent Plasticity (STDP) under 3 experimentally supported constrains: (1) specific network motif of pyramidal cells and inhibitory neurons, which was the Winner-Take-All (WTA) microcircuit motif from [Nessler et al., 2013] (2) STDP synapses between pyramidal cells and (3) a high "trial-to-trial variability".
But the presented networks needed high amounts of external input stimulation at every WTA and produced disjunct assembly groups.
So the main objective is to find networks that are less input-driven and emerge biologically plausible assembly codes. The presented approach fulfills also the 3 experimentally supported constraints:

1. The recurrent networks are based on a more biologically realistic model called Sparse Winner-Take-All (SWTA) from [Jonke et al., 2013]. SWTA has no normalization for the membrane potential and explicit inhibitory synapses, which results in a less stereotypical spiking behavior than WTAs. The lateral inhibition of WTAs is abstract modeled

(see [Klampfl and Maass, 2013] equation 3), which is basically a normalization of the membrane potential.

2. Each SWTA microcircuit is connected to other SWTAs by excitatory synapses. Every synapse is subjected to long-term potentiation (LTP) through Spike-Timing-Dependent Plasticity (STDP) and short-term plasticity (STP).

3. There is an inherent high "trial-to-trial variability", as the model is stochastic and the model parameters are drawn randomly within a range or distributed according to a probability distribution.

Using this approach our goal is to describe how to construct these networks and analyze their capability to emerge assemblies. Furthermore the following goals should be addressed:

- Smal external stimulation, to get less input driven networks
- Construct stable configurations, that are eventually scalable to easily create larger setups.
- More biological plausible assemblies that have a clear stable rank order.

The achieved results are presented in this work within the following 6 chapters:
The second chapter will present the network step by step. This will start by the theoretical basis, then define the neural parts, afterwards describe the implementation and finally the recurrent network.
The third chapter will present the used tools and methods. We will describe, assemblies, stimulus, procedures and the performance criterion to evaluate the capability to emerge assemblies.
The fourth chapter will show how to construct a stable recurrent network. At first general considerations and properties will be discussed. Afterwards a configuration is incrementally build, which will be used by all further simulations.
In the fifth chapter we will focus on different connectivity structures. We will present 3 different structures and compare their features. Afterwards the role of noise in different variation will be analyzed.
In the sixth chapter, features for biological inspired setups will be presented.

At first we will introduce a distance dependent connectivity pattern. Afterwards we will use that in a scaled up version of 100 sWTAs with more than 2600 neurons, using the previously defined model parameters.
Finally we will summarize and discuss the presented results and give a short outlook of possible future work.

Chapter 2

Model and network structure

In this chapter we will start from the theoretical model and go step by step to the recurrent implementation of the corresponding neural parts in a simulation. The basis is a biological inspired neural network model, which is called "**S**parse **W**inner-**T**ake-**A**ll" (**SWTA**) from [Jonke et al., 2013], which will be revised shortly at the first sections.

2.1 Theoretical framework

There are N observed variables $y_1, y_2, ...y_N \in \{0, 1\}$ and M hidden variables $z_1, z_2, ..., z_M \in \{0, 1\}$ and $\mathbf{W} = [w_{i,j}]_{NxM}$ encode the coupling strength between them, which could only be non-negative.

2.1.1 Prior

This model should be capable to explain an observation with multiple causes. Therefore it should be possible that several hidden variables can explain the visible data. For this reason, we use a Gaussian prior $p(\mathbf{z})$ over the unobserved variables $\mathbf{z}$ (see equation 2.1). The parameter K is the mode, which prefers K hidden causes for explaining the observations $\mathbf{y}$. A small K value therefore favors sparse activation (few causes).

$$p(\mathbf{z}) = \frac{1}{Z} exp\{-\frac{1}{2\sigma^2}(\sum_{j=1}^{M} z_j - K)^2\} \tag{2.1}$$

2.1.2 Generative model

The basic generative model is equation 2.2. The observations $\mathbf{y}$ should be explained by the given coupling strength $\mathbf{W}$ and hidden variable $\mathbf{z}$.

$$p(\mathbf{y}\,|\,\mathbf{z},\mathbf{W}) = \prod_{i=1}^{N} p(y_i|\,\mathbf{z},\mathbf{W}) = \prod_{i=1}^{N} p(y_i = 1|\,\mathbf{z},\mathbf{W})^{y_i} p(y_i = 0|\,\mathbf{z},\mathbf{W})^{1-y_i} \tag{2.2}$$

Now we define the probability for $y_i = 0$ given $\mathbf{z}$ and $\mathbf{W}$. $\overline{\mathbf{w}}_i^T$ is the i-th row of the $\mathbf{W}$ matrix which are all coupling strengths from observed variable y_i to all unobserved variables $\mathbf{z}$.

$$p(y_i = 0|\,\mathbf{z},\mathbf{W}) = e^{-\sum_{j=1}^{M} z_j w_{i,j}} = e^{-\overline{\mathbf{w}}_i^T\,\mathbf{z}} \tag{2.3}$$

For $y_i = 1$ we approximate $p(y_i = 1|\,\mathbf{z})$, where the constants a and b define the approximation-quality:

$$p(y_i = 1|\,\mathbf{z},\mathbf{W}) = 1 - e^{-\overline{\mathbf{w}}_i^T\,\mathbf{z}} \approx e^{-1/(a\,\overline{\mathbf{w}}_i^T\,\mathbf{z}+b)} \tag{2.4}$$

By plugging 2.3 and 2.4 in equation 2.2 we arrive at the final generative model 2.5:

$$\tilde{\mathrm{p}}(\mathbf{y}\,|\,\mathbf{z}) = \prod_{i=1}^{N} e^{-\overline{\mathbf{w}}_i^T\,\mathbf{z}+y_i(\overline{\mathbf{w}}_i^T\,\mathbf{z}-\frac{1}{a\,\overline{\mathbf{w}}_i^T\,\mathbf{z}+b})} \tag{2.5}$$

2.1.3 Learning

The hidden causes of an input $\mathbf{y}$ could be inferred by maximizing the log-likelihood with respect to w_{ij}. So we are using the correctly-approximated-normalized model (CAN-model) from [Jonke, 2011]. Therefore $a = 2$ and the assumption is that the generative model $\tilde{\mathrm{p}}(\mathbf{y}\,|\,\mathbf{z})$ is already normalized, so there is no need to put any normalization into the model. By going to the negative direction of the gradient, the CAN learning rule is:

$$\Delta w_{ij} = \eta z_m(y_i - 1 + \frac{2y_i}{(2\,\overline{\mathbf{w}}_i^T\,\mathbf{z}+b)^2}) \tag{2.6}$$

2.2 Network model

Now the theoretical model from chapter 2.1 should be used as a neural model. Therefore the observed variables $\mathbf{y}$ are input neurons and the hidden variables $\mathbf{z}$ are output neurons. w_{ij} is the feed-forward synaptic weight of presynaptic-neuron y_i to the postsynaptic-neuron z_j.

2.2.1 Spiking probability

According to the neural sampling theory [Büsing et al., 2011] a network of recurrently connected stochastic spiking neurons samples automatically from the posterior distribution $p(\mathbf{z} \mid \mathbf{y}, \mathbf{W})$, if the instantaneous spiking probability of a neuron j depends on its current membrane potential. So it is defined for $\delta t \to 0$:

$$p(j \text{ spikes in } [t, t+\delta t]) = \delta t \cdot \frac{1}{\tau} exp(\gamma \cdot u_j(t)) \tag{2.7}$$

Here τ is the refractory period of the neuron, which is the time after one spike, where no further spike could occur and γ is just an approximation constant.

2.2.2 Membrane Potential

The membrane potential consists of the intrinsic excitabilities α, feed-forward weights w_{ij} and recurrent inhibitory weights J_{ij}.

$$\alpha + J_{\setminus j}^T z_{\setminus j} + \sum_i^N (y_i - 1) w_{ij} \tag{2.8}$$

The rightmost term of equation 2.8 could be approximated by a constant value $\sum_i^N(-w_{ij}) \approx -const = -\delta_j$. This value could be incorporated into the prior α, so that $\alpha_j = \alpha - \delta_j$. The intrinsic excitabilities α is defined in equation 2.9 and the recurrent inhibitory weights J_{ij} in equation 2.10, which is just a constant β at every off-diagonal value.

$$\alpha = \frac{2K-1}{2\sigma^2} \tag{2.9}$$

$$J_{ij} = J_{ji} = -\frac{1}{\sigma^2} = \beta \qquad \text{for } i \neq j \tag{2.10}$$

The specific α_j is a neuron specific excitability which would also influence the inhibition through σ^2 and lead therefore to non symmetric inhibition. Therefore we use the same excitability α for all neurons. So we arrive at the membrane potential $u_j(t)$.

$$u_j(t) = \alpha + \sum_{k \neq j} z_k(t)\beta + \sum_i y_i(t)w_{ij} \tag{2.11}$$

2.2.3 Learning rule

The internal generative model (see 2.1.2) is represented implicitly through the learned weight values of each synapse. The spiking behavior of the neurons is therefore the inference step which results from integrating the synaptic inputs. The competing through lateral inhibition makes sure that only the most likely cause get's higher probability by increasing the corresponding synaptic weights. So the theoretical learning rule 2.6 has to be adapted to fit such a network model in just using local information. We exclude the non-local terms, which are not biological plausible. Therefore a local approximation is used, that emulates the optimal theoretical learning rule reasonably well in simulations:

$$\Delta w_{ij} = \eta z_m (y_i - 1 + \frac{2y_i}{(2w_{im}\,\mathbf{z} + b)^2}) \tag{2.12}$$

2.3 Neural parts

Now we have defined the theoretical basis and transformed that into a network model, which will now be implemented as neural model with explicit neurons and synapses. These will be simulated using PC-Sim [Pecevski et al., 2009], which is a neural simulator written in C++ that provides a Python interface. All neural components are implemented in C++ as neural components, calculation and data analysis were performed in Python using the NumPy and SciPy libraries ([Jones et al., 01]) and figures were created using the Matplotlib library ([Hunter, 2007]). The ZLIB framework from [Jonke et al., 2013] was used to parallelize the developed simulation scripts for the performance plots.

2.3.1 SWTA class

The SWTA-Model is defined in chapter 2.2, which we use in our configuration as basic building block. Therefore one SWTA is implemented as a Python class. Where each SWTA consist of M Exponential-Poisson-Neurons $z_0, ..., z_{M-1}$ (see 2.3.2).
Each neuron is connected to every other neuron within a SWTA through inhibitory synapses (see 2.3.4), so there are $M^2 - M$ inhibitory connections per SWTA. The excitatory external and recurrent connections could be set individually via a connector function. In chapter 2.4.2 the different connectivity structures of the recurrent setup will be further explained.

2.3.2 Exponential-Poisson-Neuron

All neurons in the setup are exponential-poisson-neurons according to [Jolivet et al., 2006]. As the firing probability has to be proportional to the membrane potential (see equation 2.7), the firing rate will be defined by the membrane potential. This results in equation 2.13. We use refractory period $\tau = 10ms$ and the constant $\gamma = 2$ for all simulations.

$$r_j(t) = \frac{1}{\tau} * exp(\gamma * u_j(t)) \tag{2.13}$$

The membrane potential from equation 2.11 is defined by the inflowing current from the synapses I_{syn} and the intrinsic excitabilities α. The excitability α (see equation 2.9) is modeled as a constant inflowing current. All other current arrives from presynaptic neurons via excitatory or inhibitory synapses. The membrane potential u_j is a voltage, so R_m has to be $R_m = 1$.

$$u_j(t) = R_m(I_{syn} + \alpha) \tag{2.14}$$

$$I_{syn}(t) = \sum_{k \neq j} z_k(t)\beta + \sum_i y_i(t)w_{ij} \tag{2.15}$$

2.3.3 Excitatory SWTA synapses

The excitatory SWTA synapse propagates current according to the alpha shaped excitatory-postsynaptic-potential (EPSP) in figure 2.1. The potential is weighted according to the local synaptic weight w_{ij}. The synaptic delay is drawn according to a Gaussian distribution where $\mu = 3ms$

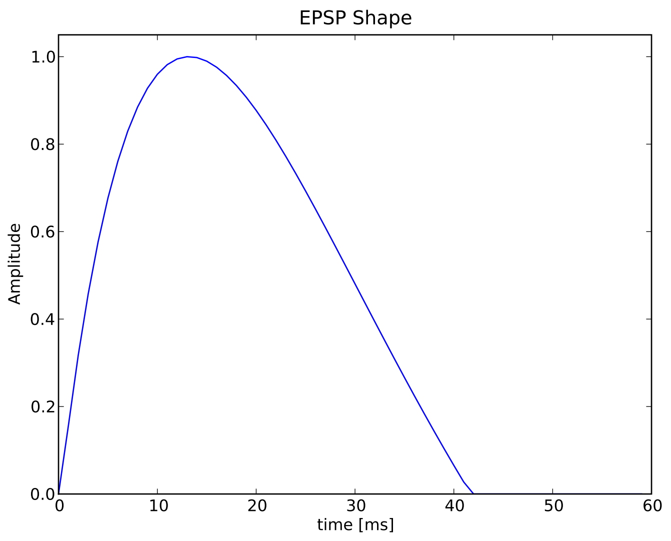


Figure 2.1: The excitatory-postsynaptic-potential (EPSP) is the potential on the postsynaptic side when a current is propagated by the excitatatory synapses.

and $\sigma = 1.5ms$. The possible weight values are in the range between $w_{min} = 0.001$ and $w_{max} = 2$. But the initial weight is randomly drawn according to a uniform distribution between $w_{min} = 0.001$ and 1. This weight is updated according to the learning rule from equation 2.12, which is now implemented as follows:

$$\Delta w_{ij} = \eta * (w_{ij} * f - 1)$$

$$f = 1 + \frac{a}{(aw_{ij} + b)^2}$$

Because we are using the CAN SWTA model, the approximation constants are $a = 2$ and $b = 0.8$. In all presented simulations we are using the learning constant $\eta = 0.05$.

2.3.4 Inhibition-Synapses

The lateral inhibition $\beta = \frac{1}{\sigma^2}$ is implemented by current based static spiking synapses from each neuron to all other neurons within the SWTA. Therefore there are $M^2 - M$ inhibitory synapses.

When the presynaptic neuron spikes, then there is current propagated

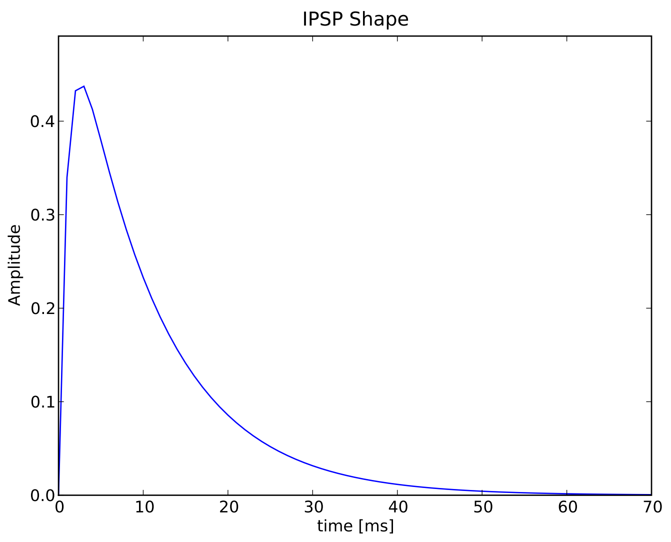


Figure 2.2: The inhibitory postsynaptic potential (IPSP) is the potential at the postsynaptic side from an inhibitory synapse. The synaptic delay is drawn according to a Gaussian distribution where $\mu = 2ms$ and $\sigma = 1ms$. If a neuron spikes all other neurons within the SWTA are inhibited with that potential. The maximum amplitude is defined by the amount of inhibition, which is $\beta = \frac{1}{\sigma^2}$

through the $M - 1$ inhibitory synapses to the postsynaptic neurons. The postsynaptic potential is the double exponential shaped IPSP curve from figure 2.2.

2.4 Recurrent network configuration

In the previous section we have defined all elements necessary for a feed-forward implementation. However for a biological inspired recurrent implementation, there is the need for some additional parts.

2.4.1 Network class

All SWTAs are placed on a 2D grid, where each grid point is represented by one SWTA class instance (section 2.3.1) with a random number of hidden neurons. The simulation procedures and data handling is implemented within this class. In chapter 4, we are using a $3x3$ grid of 9 SWTAs and in chapter 6 a $10x10$ grid of 100 SWTAs within a range from 24 to 30 neurons.

2.4.2 Connectivity patterns

Excitatory connections are either from an external poisson neuron or from a neuron of another SWTA. There are 4 different motifs for connecting external channels, which will be explained further in chapter 5.
The internal recurrent synapses connect one neuron of a SWTA to another neuron from another SWTA according to a probability. In chapter 4 we are using a uniform distribution, where for example a neuron connects to another neuron with probability $p_{conn} = 0.6$. In chapter 6 we are also using a distance dependent connectivity pattern, which will be further explained in chapter 6.1.1.

2.4.3 Dynamic synapse

All excitatory synapses within the recurrent network are subjected to short-term and long-term plasticity. The long-term plasticity (LTP) is the adaption of the weights by the learning rule 2.3.3.

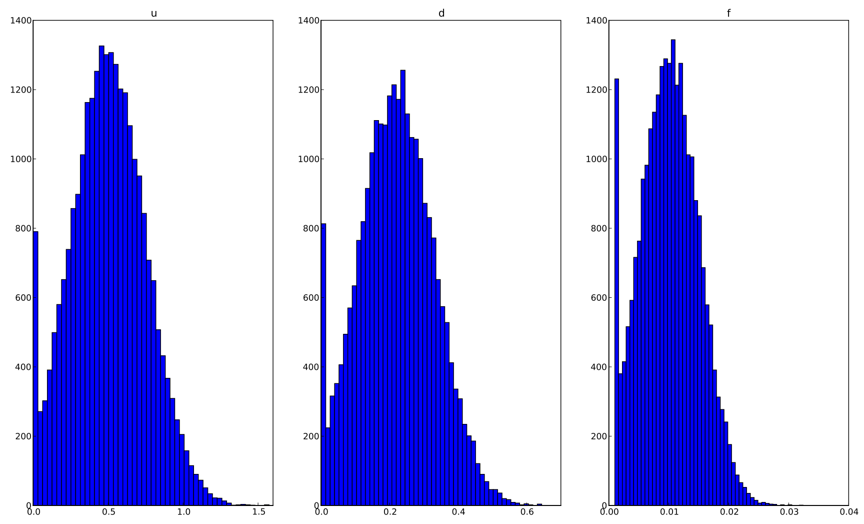


Figure 2.3: The UDF-Values of the dynamic synapses are spread according to a truncated Gaussian distribution, where all values below zero are set to zero. **left:** The available Potential u has a mean of $\mu_u = 0.5$ and $\sigma_u = 0.25$ **middle:** The Gaussian distribution of the depression time constant d has a $\mu_d = 0.22$ and $\sigma_d = 0.11$ **right:** The facilitating time constant f has a $\mu_f = 0.01$ and $\sigma_f = 0.005$

For the short-term plasticity (STP) a model based on [Markram et al., 1998] is used. The u, d and f values are chosen independently for each synapse according to a truncated Gaussian distribution (see figure 2.3). These distributions are based on data from [Markram et al., 1998].

Chapter 3

Determining self organization

In this chapter we will describe the kind of self organization we are looking for and which procedures, tools and method will be used.
At first we will define the network organization called assembly. Afterwards we will describe the stimulus and the simulation procedure. At last we will define the tools for analyzing the trained networks, including a performance criterion for evaluating the capability to emerge assemblies.

3.1 Cell assemblies

Groups of neurons that are coactive for a stimuli are called cell assemblies [Hebb, 1949]. They play a key role in neuroscience for neural computations. We use the definition from [Buzsáki, 2010], where a structured activity of a specific subset of neurons to a repeated input pattern is an assembly code. The main part of this work is to analyze the capability of different networks to emerge such assembly structures and to observe how different structures influence these capabilities.
As this model is biologically inspired, we expect that the assemblies should resemble biologically observed data like [Harvey et al., 2012], where for example neurons are part of more than one assembly.

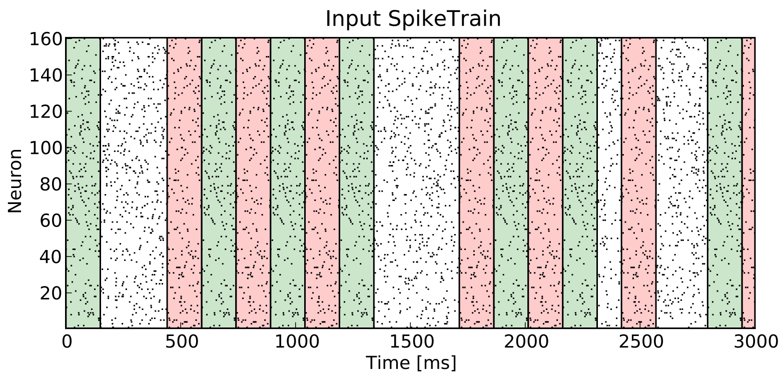


Figure 3.1: **Stimulus**: a typical stimulus of 160 Poisson spike trains at constant rate $r_{in} = 10Hz$. Two frozen $150ms$ spike patterns (green and red shaded areas) are embedded at random intervals in the poisson spike trains. After a pattern is shown, there is a probability of 0.5 to show noise. The noise length is randomly drawn between $50ms$ and $450ms$

3.2 Stimulus: external spike train

The external input is a frozen spike time pattern of $T_{Pattern} = 150ms$ length, which is embedded in uncorrelated Poisson spike trains (see figure 3.1). These patterns are shown at random times, where the rate r_{in} of all channel is constant, regardless if there is noise or the pattern present. After each frozen pattern sequence, noise is shown with a probability of 0.5. The Poisson spike train length is randomly drawn between $50ms$ and $450ms$. The network should respond to each input pattern with a specific temporal activation of a subset of neurons.

3.3 Simulation procedure

The same procedure is followed for all presented configuration, to be able to easier compare the result. All presented configurations are learned for 100s. In the learning phase the synaptic weights are adapted to the CAN learning rule. Afterwards the learned Network is tested for 100s, where the synaptic weights are fixed. There is a response test for the pattern also before learning, to show that the assembly structure emerged only while learning. Both spike trains will be plotted for each structure. Each simulation is simulated at a time step of $dt = 1ms$.

3.4 Determining and evaluating assemblies

To search the parameter space there has to be a defined measurement for the capability of a configuration to emerge assemblies. For doing that we use the precision measurement to select the most precise neurons and form a group signal of them. Afterwards the correlation coefficient of this group signal and a pattern signal is calculated, which is used as a performance criterion.

3.4.1 Precision measure

We define two binary activity signals $\mathbf{a}_j[t] \in \{0,1\}$ for the neuron activity and $\mathbf{p}_k[t] \in \{0,1\}$ for the pattern activity, according to the the neural sampling theory [Büsing et al., 2011]. So both signals are one for the period τ

after one spikes, which is $\tau = 10ms$ for all simulations.
Therefore the activity signal for the neuron j at time t is $a_j[t]$, which is one for the period $t =$ [spike, spike + τ].
The pattern activity $\mathbf{p}_k[t]$ is defined in the same kind, while the spike time pattern k is shown $p_k[t] = 1$ for $t =$ [pattern_start, pattern_end + τ]. As $T_{pattern} = 150ms$ in all simulations, there will be ones for $160ms$ for each pattern representation.
We will measure the precision $prec_{j,k}$ for each neuron j to pattern k according to equation 3.1. So we are actually calculating the fraction of correct spikes, to the sum of all (correct and false) spikes.

$$prec_{j,k} = \frac{\sum_{t=0}^{T} a_j(t) p_k(t)}{(\sum_{t=0}^{T} a_j(t) p_k(t) + not(a_j(t)) * p_k(t))} \tag{3.1}$$

3.4.2 Assign neurons to an assembly

We are looking for a subset of neurons that respond to a specific input stimulus. So we assign the most precise neurons j regarding pattern k to the assembly A_k. Therefore we set a threshold value $thresh$ and find all neurons with a precision measure of $perf_{j,k} > thresh$ to the assembly group A_k. We used a precision of $thresh = 0.4$ for our simulation.

3.4.3 Assembly group correlation measure

Now we are evaluating the performance by how accurate an assembly group is in regard to the shown pattern. We do that by looking at the correlation of the found assembly A_k to the pattern activity $\mathbf{p}_k$. The activation signals $\mathbf{a}_j[t]$ of all neurons in an assembly A_k is then used to form a group signal $\mathbf{g}_k[t] \in \{0, 1\}$, by combining all spikes of the activity signals $\mathbf{a}_j[t]$ of an assembly A_k. Afterwards we calculate the correlation coefficient $perf_k$ of this assembly group signal $\mathbf{g}_k$ and the pattern signal $\mathbf{p}_k$. This correlation measurement is used as a performance criterion for evaluating network configurations.

3.5 Assembly activation trajectory

Having determined the assembly, we order the neurons according to their mean activation during an input pattern representation. This order represents the activation sequence of a memory trace regarding the presented input.

3.5.1 Mean activation

The mean activation is calculated according to the mean spike latency in [Luczak et al., 2009]. For each neuron j, we considered the temporal activity during each pattern representation of a particular input pattern. We define as pattern representation time $T_{repr} = T_{pattern} + 2\tau$. In all our simulations $T_{pattern} = 150ms$ and $\tau = 10ms$ and therefore the representation time will be $T_{repr} = 170ms$.
At first we calculate the corresponding rate $r_j(t)$ for $t = 1, ..., T_{repr}$. Now we are taking the mean of all pattern representations at each time point t. The mean rates are plotted as mean activation, for example in figure 4.2 on the left side. But in these plots only higher active neurons above a minimum rate r_{min} are shown.

3.5.2 Ordering: mean activation time

The neurons within an assembly are ordered by their center of mass t_j^* according to [Luczak et al., 2009]. The center of mass t_j^* is calculated according to equation 3.2, where time is interpreted as angle in the complex plane. We compute the mean complex number weighted by the activation $r_j(t)$. So the ordering of the neurons within an assembly is by ordering their mean t_j^* values. Which are the white dots in each plot of the mean activation (see for example figure 4.2).

$$t_j^* = \frac{T_{repr}}{2\pi} \cdot arg \left[\frac{\sum_{t=0}^{T_{repr}} r_j(t) exp(\frac{t}{T_{repr}} 2\pi i)}{\sum_{t=0}^{T_{repr}} r_j(t)} \right] \tag{3.2}$$

Each plotted spike train is also ordered to this mean activation time.

3.5.3 Rank order correlation

We want to quantify how reliably the order of activations are. This is measured as Spearman's rank correlation between two rank variables. The followed procedure is according to [Luczak et al., 2009], where we calculated histograms of rank correlations between each single trial and the mean activation. Rank order correlation indicates the reliability of the learned activation trajectory, which ranges from -1 to +1. So positive correlations indicates a maintained order of activation during different trials.

Chapter 4

Learning behavior of recurrent configurations

This chapter will describe the main considerations and parameters for building a stable network configuration. This will be presented by an incrementally build configuration, which we will use for all further simulations.
As there have to be many different properties and parameters considered, we will start with general dependancies and assumptions. Afterwards we will incrementally specify parameter ranges for the most important parameters. At the end of this chapter the model will be simulated and analyzed with the initial structure (setup 0).

4.1 General important issues

This section will describe the most crucial aspects, which could be the cause for serious problems in finding a stable setup.

4.1.1 Spontaneous activity

As the learning rule is a form of Spike-Timing-Dependent Plasticity (STDP), there have to be spikes present to explain the input. If there are no spikes, there is no possible explanation and nothing is learned. But if there are too many spikes, the explanation is not very useful; rising all these weights could even result in bursting. Therefore spontaneous activity is crucial for learning:

- Using a too low excitability (very negative K) or high inhibition (very small σ^2) results in virtually no spontaneous activity
- Using a higher excitability (low negative K) with too less inhibition (bigger σ^2) does lead to spontaneous activity, but could result in bursting.

Choosing these parameters accordingly is a crucial part. But these parameters change also with the size of the network, for example if there are more neurons within a SWTA, then more inhibition signals will lower the membrane potential of each single neuron. Therefore a lower inhibition β or higher excitability α will be needed.

4.1.2 Normalization

As the SWTA model assumes that the generative model $\tilde{p}(\mathbf{y} \mid \mathbf{z})$ is already normalized (see 2.1.2), there is no normalization within the model. So one cannot inject arbitrary input or use arbitrary input/recurrent-ratio for connecting, because the input strength has an immediate effect on the activity regime of the network. This is particularly challenging because recurrent learning constantly changes how strong the network will respond to a given input. Therefore it is essential to examine, how to keep the network in a working regime. This is influenced by:

- external input: number of input channels $N_{Input,External}$, external input rate r_{in}
- amount of recurrent connectivity: only number and connectivity-motif of recurrent connections
- right amount of inhibition β and excitability α to keep the membrane potential u_j in a stable region.

The structural issues directly influence the membrane potential for example (1) more neurons leads to more inhibition signals which lower the membrane potential (2) more connectivity rises the possible excitation, which could push up the membrane potential very fast and result in bursting.

4.1.3 Possible approaches

For resolving problems with the above mentioned issues one has to chose one of the following paths:

1. Adapting the excitability α_j (see equation 2.11) per neuron z_j to control the membrane potential u_j, by using some additional mechanism, for example Feed-Forward Inhibition [Keck et al., 2012], Homeostasis [Habenschuss et al., 2013], or similar.

2. Constrain the structure, so that only a specific amount of input stimulation is possible and the structure; how the stimulation gets into the network is specified.

We have decided to work with the "pure" SWTA model and analyze the behavior of different structures. So this thesis deals mainly with structures and the self organization. We will specify one model and analyze the behavior of different structural constraints. Therefore we will consider different network architectures that allow an excitability that supports spontaneous activity and does not lead to bursting. We use the same model parameter ranges for all SWTAs within the network.

4.2 Single SWTA example

Now we will start with a single feed-forward SWTA and analyze the learning behavior. According to [Zeno, 2012]), a single SWTA is capable to learn the temporal structure of patterns. The key issue is to control u_k, which could be done by tuning two properties: (1) a right amount of stimulation (input spikes for a neuron) and the (2) right amount of competition (lateral inhibition). Because of this competition, neurons are forced to specialize for a specific time segment.

4.2.1 Input stimulation

In all presented configuration there are approximately 64 external input channels, where the frozen spike time pattern is presented. Each channel has a constant rate $r_{in} = 10Hz$. Therefore there are approximately 6400 spikes per $Neuron/s$ from external sources. The amount of evidence needed for learning could be influenced with the excitability $\alpha = \frac{2K-1}{2\sigma^2}$. We will adapt the excitability by changing the parameter K (number of preferred causes, see 2.1.1).

4.2.2 Competition

After a neuron emits one spike, it inhibits all other neurons within the SWTA. This lowers the probability of other neurons to spike in the same time segment. Too much inhibition (low σ^2) damps the dynamic very much, too little inhibition (high σ^2) let's more neuron spike for the "events with most evidence", which could result in bursting. We will control the inhibition β mainly by using the parameter σ^2 ($\beta = \frac{1}{\sigma^2}$). This has shown to be a good approach to balance u_k, as σ^2 is also part of the excitability α.

Assembly after 100s learning

After 100s a feed-forward SWTA with $N = 100$ neurons has learned a stereotypical activation trajectory for a specific input. We see that neurons specialize for specific time points and that their activation order is positive correlated in figure 4.2. The formed assembly has a correlation of 0.64 and consists of 57 neurons. 43 neurons stay free. In figure 4.3, we see the spike train of the neurons before and after learning. They are ordered by their mean activation time point within the assembly.

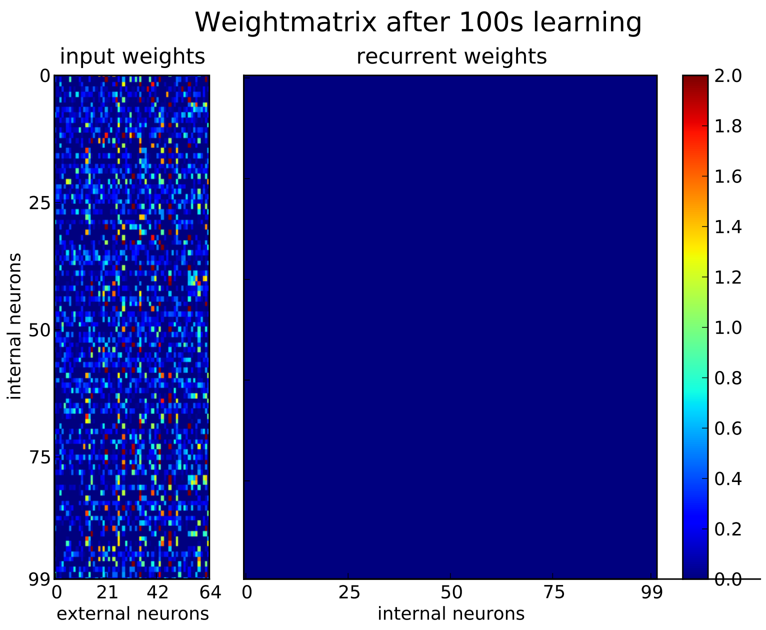


Figure 4.1: **left:** The synapses are connecting the external input neurons on the horizontal axis (presynaptic) to the postsynaptic SWTA neurons on the vertical axis. **right:** As this is just one feed-forward SWTA with 100 neurons there is no recurrent connectivity.

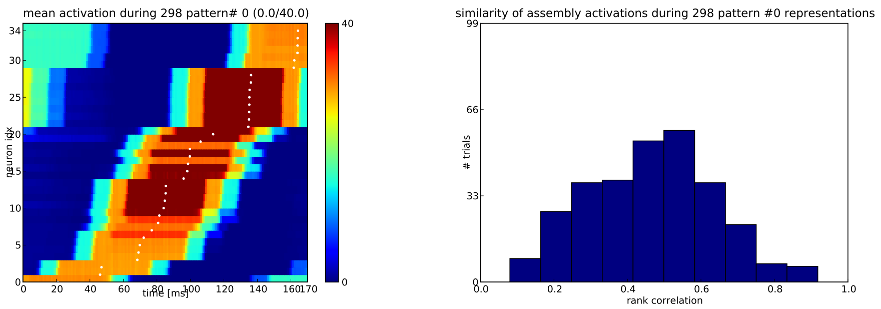


Figure 4.2: **left:** Average activation of each neuron in the assembly during 298 representations of the pattern. The neurons are sorted by their mean activation time which is represented by the white dots. **right:** Histogram of rank correlation between mean activation times for each pattern presentation.

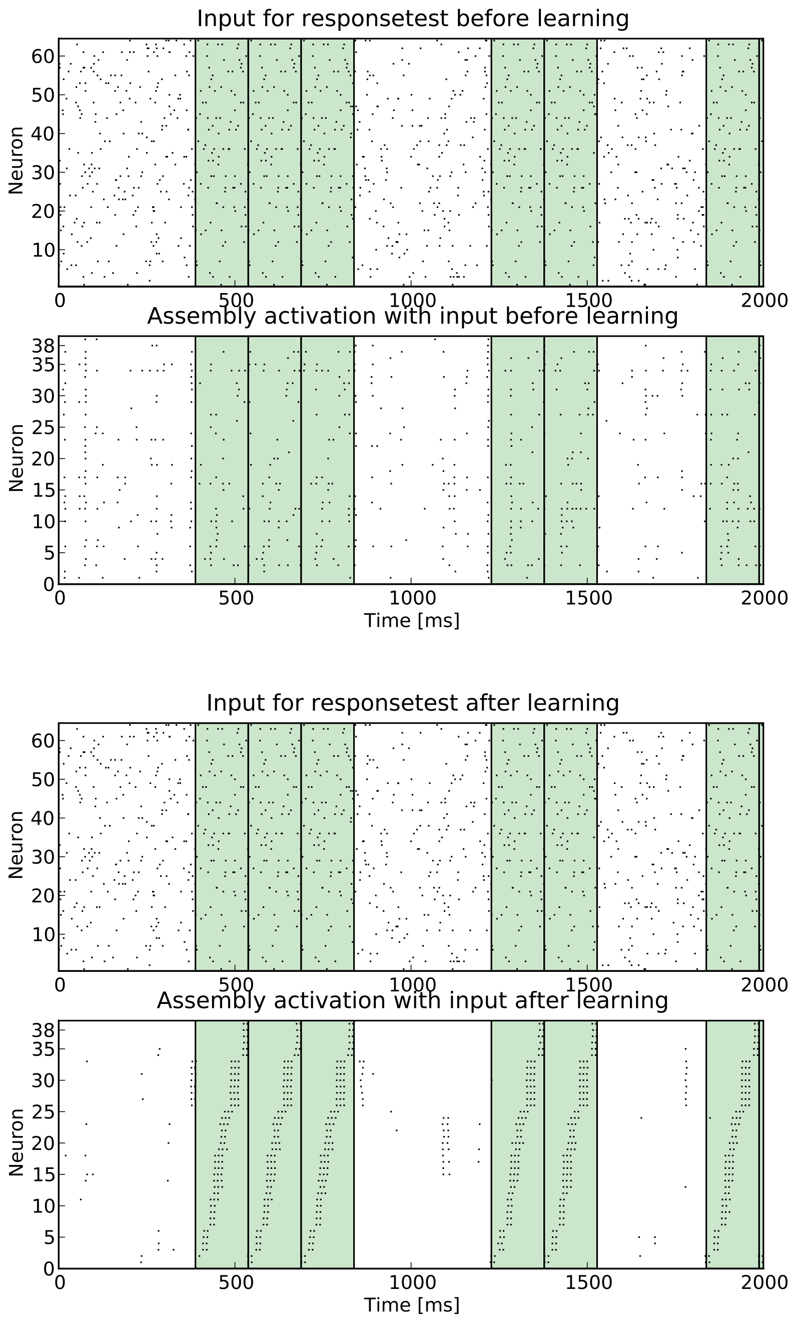


Figure 4.3: Single SWTA response to input before and after learning. All internal neurons are sorted by their order of activation within the assembly: **top:** input sequence to test the untrained network and below the corresponding output of the untrained network. **bottom:** input for testing the trained network and below the spike train of the ordered assembly neurons: there is a clear activation trajectory for the frozen spike-time-pattern and very little response while noise is shown

4.2.3 Missing capabilites of the single SWTA setup

The number of neurons sets the maximum possible time points in which neurons could specialize. In order to form bigger assemblies or more different activation trajectories, there have to be more neurons. Each neuron is subjected to the same amount of inhibition β. So the inhibition and has to be adapted to the size. So using more SWTAs of similar size, overcomes this need for adaptions. In the later setups, we will use randomly generated SWTAs with same parameters K and σ^2, which dramatically reduces the need for parameter optimization, then in section 6.2 we will use the same model parameters for 100 SWTAs.
Furthermore this setup is only input driven, because there is no internal dynamic. This will be different for the recurrent networks in the next chapters.

4.3 Specifying the recurrent setup

In this chapter we will specify the SWTA model parameters, which we will use in all further simulation to better compare all different structures. All presented configurations are based on a $3x3$ grid of 9 SWTAs with the parameter ranges, defined in this section. We are using a uniform connection pattern, because a $3x3$ grid is too small for distance dependent connectivity. The external input channels will be connected according to Setup 1 (see chapter 5.1), because the initial setup is not very capable to learn more assemblies (see 4.4.2).

4.3.1 Number of hidden neurons N

Now we are using nine SWTAs in a $3x3$ grid, where each resembles that of chapter 4.2 and has exactly the shown number N of hidden neurons $\mathbf{z}$. A uniform recurrent connectivity pattern is used, where each neuron is connected to another one with the probability $p_{rec} = 0.6$.
In figure 4.4 is the mean performance and standard error of 100 trials. We see that 20 neurons is the minimum for a good performance. Learning 2 patterns with more than 32 neurons decreases slightly the performance. There are multiple possible explanations: (1) The assemblies get much bigger and this decreases the correlation as there are more spikes at wrong times. (2) The uniform connection is based on the number of neurons, which will increase the number of recurrent connections by the same amount of external input connections. (3) The model parameters have to be adapted for this bigger model, as more neurons emit the same amount of inhibition, the membrane potentials are lower (see section 4.2.3).
Further experiments have shown, that the performance is further increased by using different number of neurons. Therefore we will choose a randomly drawn number of neurons in the range from 24 to 30 neurons, as this is a good range in figure 4.4.

4.3.2 Recurrent connectivity p_{conn}

In this section we will observe the influence of different recurrent connection probabilities. We are using the configuration from the previous section

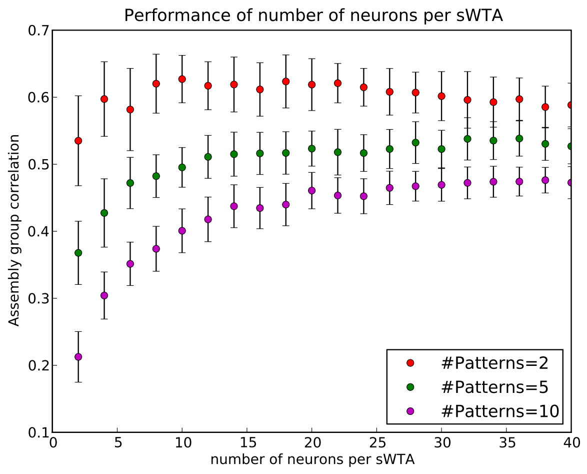


Figure 4.4: Performance of different neuron numbers N, over 100 trials with standard error. The optimal range is above 20 neurons. Above 32 neurons the performance of learning 2 patterns starts to slightly decrease.

with the specified randomly drawn number of neurons between 24 and 30. Each configuration is simulated for 100 times and the mean performance and standard error is plotted in figure 4.5.

We expected that a very high amount of recurrent connectivity will prevent learning, as the huge amount of connections will be a problem for the membrane potential. But the surprising result was, that the amount of recurrent connections did just slightly influence the performance. For learning 2 patterns it did not make much difference to have 25% (766 recurrent connections) or 95 % (42538 recurrent connections).

The performance drop above 0.7 while learning 10 patterns is probably due to the lack of capacity, as there are additional internal signals learned. This could probably overcome by using more SWTAs or just neurons, because we are using a mean of 243 z-neurons for 10 different 150ms patterns, whereas each neuron should specialize on one of 1500 possible time points.

Further analysis indicated that if the network was in a working regime with high connectivity, it decreased a lot of "unneeded connections" within a short time and came to the same results as in a more sparse initial setting. This was even possible in a nearly unstable configuration, where the network starts bursting when injecting uncorrelated noise instead of the pattern.

As there is no clear optimum, we choose $p_{rec} = 0.6$ for all future simulations.

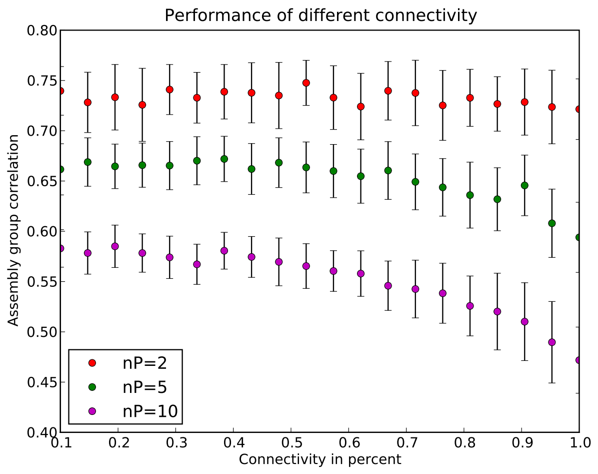


Figure 4.5: Now a randomly drawn number of z-neurons between 24 and 30 was simulated for 100 trials. The dots are the mean and the bars resembles the standard error. The surprising result is, that the amount of initial connections has not a real influence for less than 5 patterns. Larger number of patterns decreases the mean performance and if there are many initial recurrent connections then the performance is further decreased.

4.3.3 Excitability α by chosing K

Choosing an appropriate K value is important as it directly influences the amount of spontaneous activity (see section 4.1.1) and how much "evidence" is necessary to specializate a neuron. We show here the mean performances and standard error of different excitabilities, using a random configuration according to the previous defined parameter ranges. The excitability α is adapted by changing K and leaving $\sigma^2 = 0.39$ constant.
In Figure 4.8 we see that the performance is mainly influenced by the excitability. We have to use a value between -11 and -7 to achieve good results with this inhibition value β. We see that this configuration is capable of learning more assemblies, even with 5 assemblies a mean group correlation above 0.6 is possible. For all further simulation we chose $K = -9$

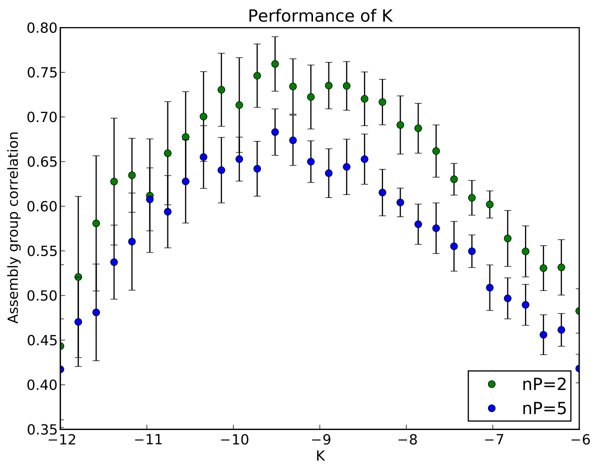


Figure 4.6: Performance of different excitabilities by using different K values and same $\sigma^2 = 0.39$ over 30 runs to learn 2 and 5 patterns, with standard error. Choosing the K value has high impact on the performance.

4.3.4 Adapting the competition β with σ^2

Now we are varying the amount of competition by changing σ^2. The competition is the key for successful learning, as it directly influence the inhibition to all other neurons within a SWTA and the excitability α. So it controls the activity of a SWTA and how many neurons could specialize for specific time points.

In figure 4.7 we see that there is only a small range of σ^2-values for achieving good performance. This region is directly connected with the chosen K value through α (see equation 2.9). We choose $\sigma^2 = 0.384$.

4.3.5 External stimulation r_{in}

To compare the different structures we want to have the same external stimulation. We are using the same input stimulus as in 4.2.1. To keep the number of external channels constant we are using 160 channels and connect each one with a probability of $p_{conn,input} = 0.4$. Therefore each SWTA has a completely different external input with similar number of channels. Each channel provides the same rate r_{in}, which is crucial as it provides the amount of evidence for learning. Too little stimulation wouldn't suffice to

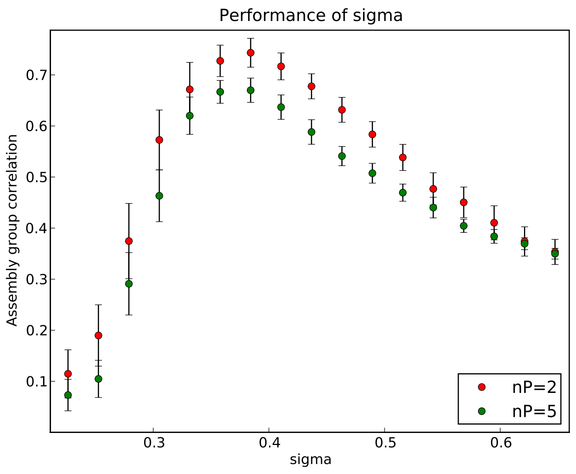


Figure 4.7: Performance of different inhibition values by using different σ^2 over 100 runs with standard error.

trigger successful learning and too much stimulation would trigger too much activity, which results in a bad performance (see section 4.1.1).

In figure 4.8 we see the performance of different rates by using the previously defined model parameters. We have now 160 channels with the specified rate r_{in}, where 40% are connected to each SWTA. To achieve a performance above 0.5, we have to use at least 9 and at max $10Hz$. Which means that there should be a input stimulation of more than $5120 spikes/s$ and below $7040 spikes/s$ for that kind of stimulus. To overcome that constraint, one has to adapt the excitability α or even the learning rule, which would be a major model modification (see 4.1.3). Nevertheless we will use $r_{in} = 10Hz$ for future simulations.

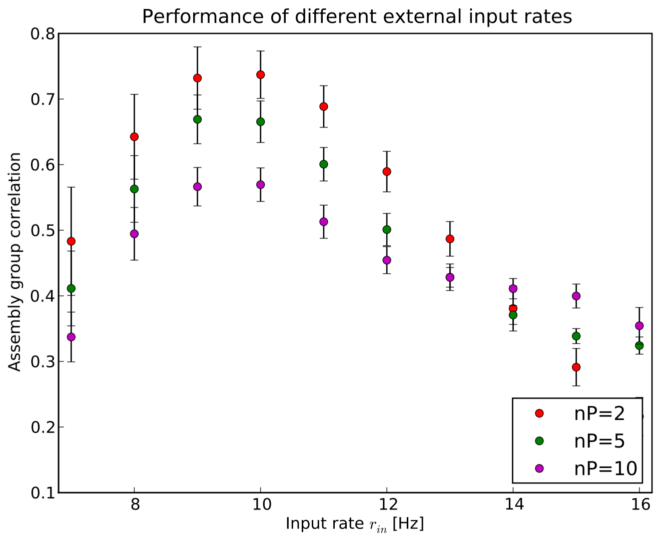


Figure 4.8: Performance of different external input rates r_{in} to learn 2, 5 and 10 patterns over 100 trials with standard error. The external input stimulation determines the amount of "evidence" for the pattern, so it directly influences the learning performance.

4.4 Initial configuration (setup 0)

In the previous section we have defined the most important model parameters, which we will use now in a simple structure, where the external channels are connected to all neurons (see figure 4.9). We will use the defined procedures and tools from chapter 3 to analyze the stimulus specific assembly structures.

4.4.1 Results after 100s learning

In figure 4.10 we see that nearly all neurons have high synaptic weights for the same external input channels. The histogram below also shows that input weights are spread over the whole weight values, whereas the recurrent connections are nearly all below 0.5. These high weighted channels are the cause for bursting, when multiple spikes occur in these channels.
This could be seen at the mean activation plot 4.11, where the whole first assembly spikes at the same time. The second assembly has a short activation trajectory after the burst, but has the same problem.
We tried to optimize the model parameters by adapting K, σ^2 and the amount of connectivity. The above mentioned effect is present in all simulated configurations, it was just possible to slightly reduce it.
One of the best simulations is plotted in figure 4.12. This configuration is only capable to emerge one assembly, because further assemblies only consists of many neurons spiking at nearly the same time (bursts).
This simulation had a mean performance of 0.68. The first assembly is quite good with a correlation coefficient of 0.76 and 121 neurons, but the second with a good correlation of 0.6 and 90 neurons is very concentrated at the end of the presentation time. There are 9 neurons active for both activation trajectories, but no free neurons.
So we concluded that the bursting behavior is caused by this kind of connectivity structure.

4.4.2 Problems regarding the initial setup

- The weights adapt very fast to the highly correlated input. All recurrent weights go down below 0.5 at a very short time-scale. We see that

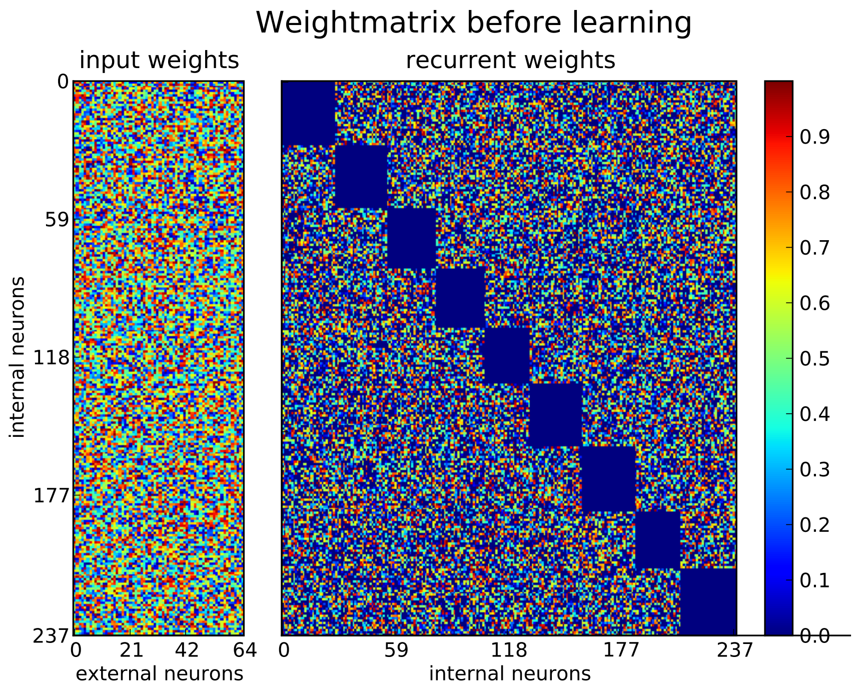


Figure 4.9: **Initial setup 0**: horizontal axis are the presynaptic neurons and vertically are the postsynaptic neurons. Each recurrent and input weight is initialized uniformly between $w_{min} = 0.001$ and 1. The maximum of each weight after learning is $w_{max} = 2$.
input weights: there are 64 external channels, each channel connects to each neuron. In total there are 15232 input connections (33.4%).
recurrent weights: each neuron connects with each other neurons with a probability of 0.6, which results in 30419 connections. This is a mean of 128 (min=110 max=143) connections per neuron

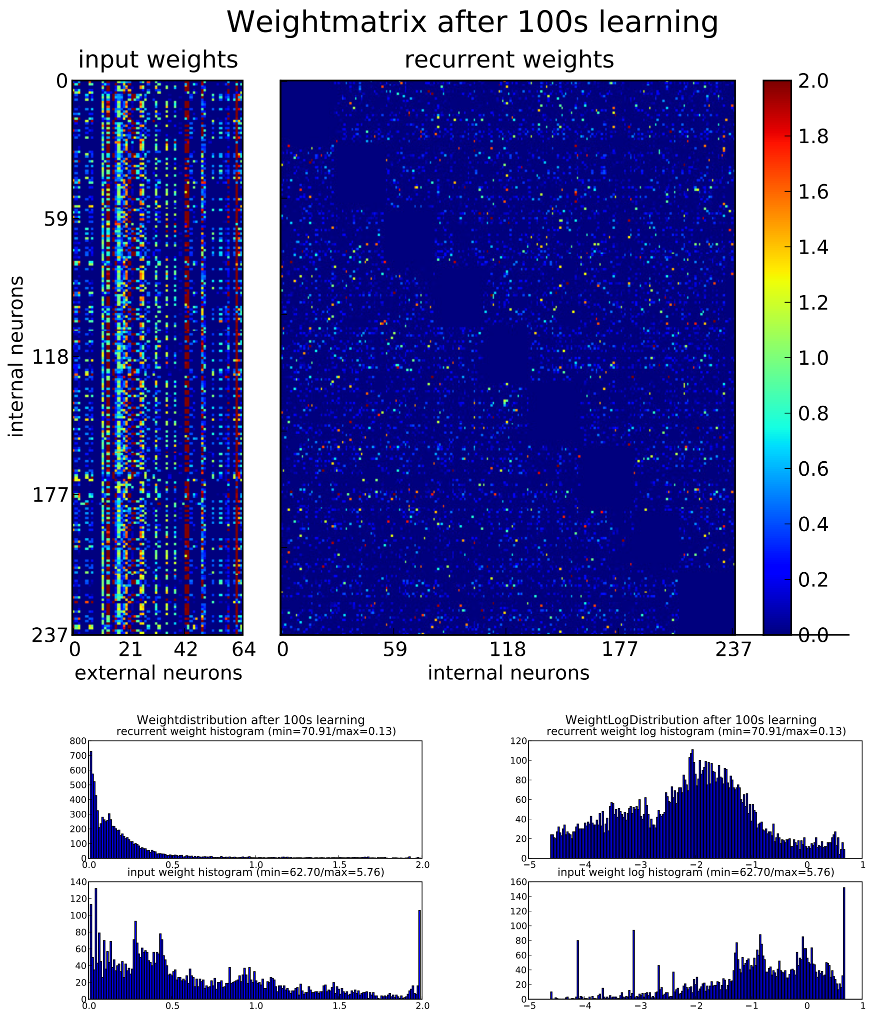


Figure 4.10: Setup 0 after 100s learning: Some external input channels are at maximum weight $w_{max} = 2.0$ for nearly all neurons. That leads to very high stimulation when this channel is on, which could result in bursting. **bottom**: Histogram of weights on linear- and log scale: The most recurrent weights go down below 0.5, the input weights spread over the whole possible range.

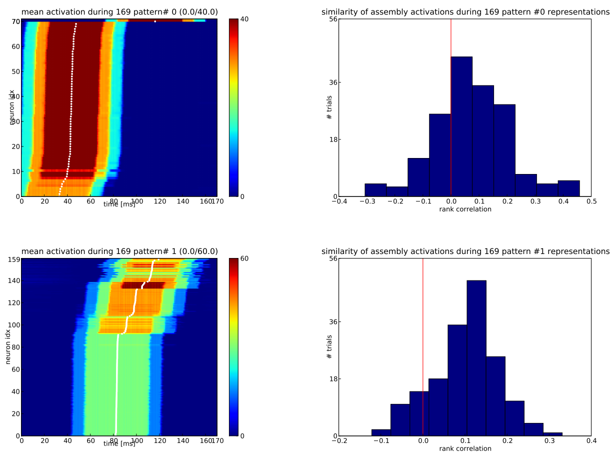


Figure 4.11: Mean activation and rank correlation using parameters from section 4.3: The high weights for the same input channels causes bursting. No activation trajectory is spread to the whole presentation time.

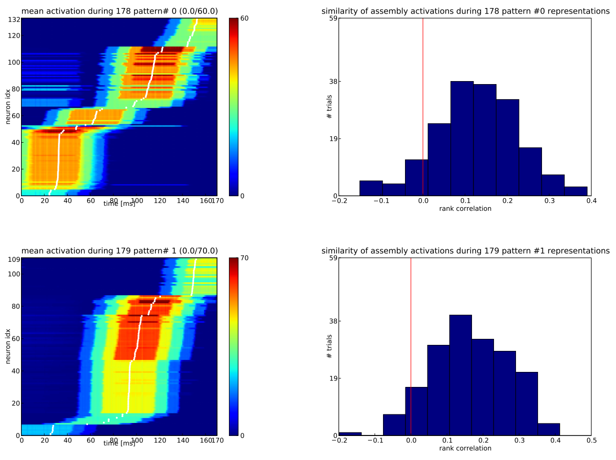


Figure 4.12: Mean activation and rank correlation using **optimized parameters:** one assembly is learned, but both assemblies consist mainly of bursts.

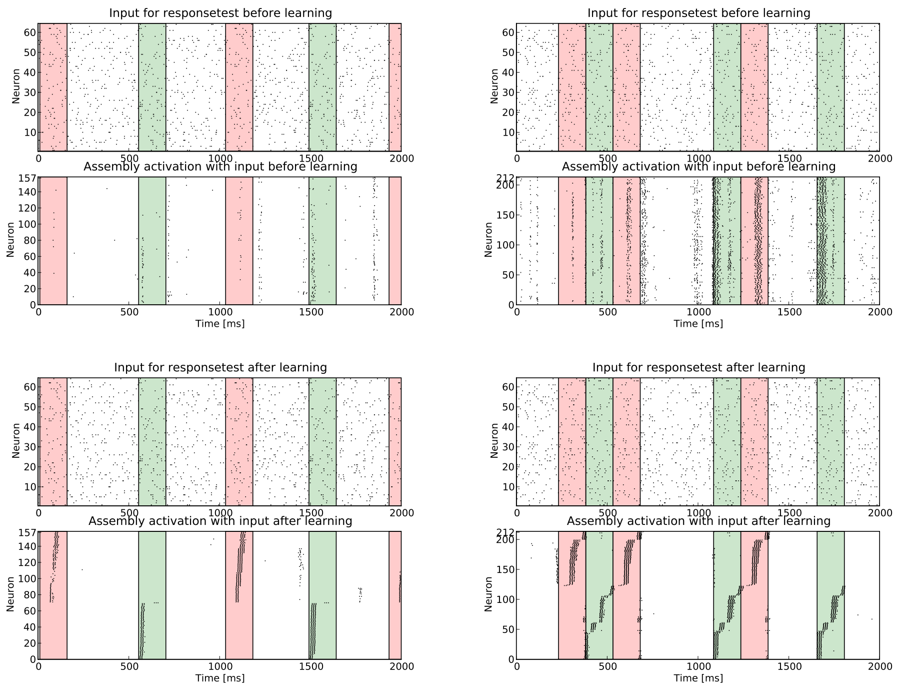


Figure 4.13: **Left column** simulation with the parameters from section 4.3. Both assemblies are mostly a burst at the same time **right column:** Optimized parameters for setup 0: We see that an assembly structure emerge, but only 1 Assembly is learned with a clear activation trajectory, the second is mostly a burst at the same time.

very few recurrent connections survive and that specific input channels have high weight values for all neurons, which facilitates bursting.

- This setup is not capable of learning more than one assembly. At best only the first assembly has a clear activation trajectory, the second is mostly a burst at the same time. Also the size of the first shown assembly is larger than the second one and there are no free neurons left after a simulation. So the capability of learning more input specific trajectories is substantially reduced after each presentation.

Chapter 5

Different recurrent connectivity structures

In the last chapter we have defined parameter ranges for the SWTA model and simulated the initial structure. The first recurrent setup had some serious drawbacks. The conclusion from the initial setup simulations was that providing all neurons with the same input leads to too high correlation for the input channels, which results in a very input-driven network. We want to have a recurrent setup, which is capable to emerge assembly structures and is less input-driven. For achieving that, we propose 2 different approaches, which results in the following 3 setups:

1. Setup 1: Provide each SWTA with a different external input.
2. Setup 2: Generate internal activity.
3. Setup 3: Combine both approaches, to have internal activity and different external input.

In this chapter a typical example for each configuration is shown and the resulting assemblies are analyzed with the methods from section 3. Afterwards the role of noise is further examined.

5.1 Setup 1: different external input

With this structure we want to provide each SWTA with a different kind of external stimulus, but use the same SWTA model from the last chapter. So we have to keep the external stimulation equivalent (compare to section 4.2.1). Therefore we need a mean of 64 external input channel per SWTA with the same rate $r_{in} = 10Hz$.
The solution is, that each SWTA receives a random fraction of 160 external input channels with a probability of 0.4 drawn for each channel.

5.1.1 Learn 2 patterns using setup 1

We show here a typical example for setup 1. We are using the above mentioned external connectivity motif and the defined model from the previous chapter to learn 2 spike time patterns.

The randomly generated network structure is shown in figure 5.1. We have a mean number of 191 presynaptic connections per neuron, where at mean 62 channels (min=44, max=73) are external, which is a fraction of 32%.

The network formed two input specific activation trajectories which have both very high correlation to the stimulus and are spread over all 9 SWTAs. The first assembly has a correlation of 0.81 and consists of 63 neurons (26%). The second assembly has a correlation of 0.76 and consists of 114 neurons (47%). There are 68 free neurons (28%) and 4 overlapping neurons in 3 different SWTAs. They are plotted twice in figure 5.4, as they have different rank orders in different assemblies.

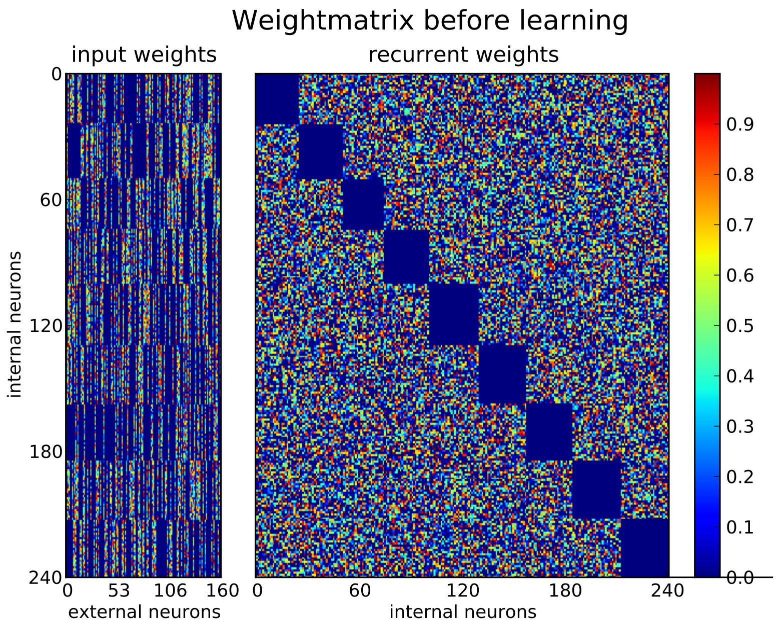


Figure 5.1: **Setup 1 initial synaptic weights**: horizontal axis are the presynaptic neurons and vertically are the postsynaptic neurons. Each recurrent and input weight is initialized uniformly between $w_{min} = 0.001$ and 1. **input weights**: there are 160 external channels, each channel connects to a SWTA with probability 0.4, this results in a mean of 62 (min=44, max=73) connections per neuron. In total there are 31177 recurrent connections and 14954 input connections. So each SWTA receives a mean of 32.3% external input stimulation. **recurrent weights**: SWTAs are all-to-all connected, each neuron connects with each other neuron with a probability of 0.6, which results in a mean of 130 (min=110, max=144) connections, which is 67.3% of the stimulation.

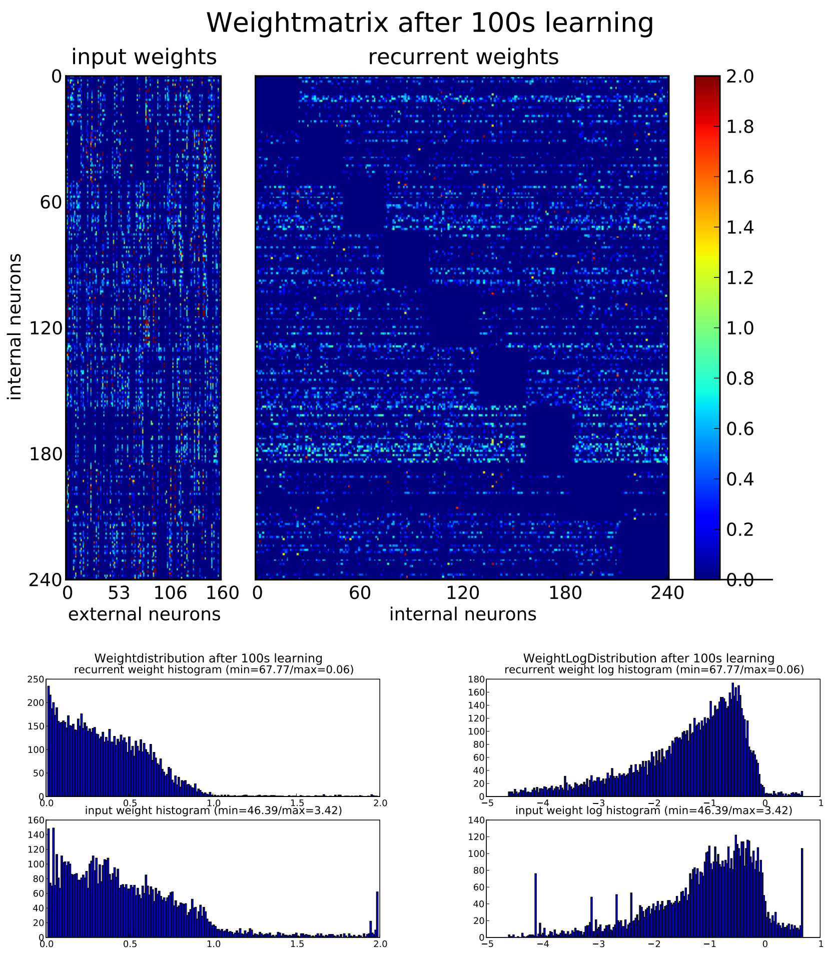


Figure 5.2: Setup 1 synaptic weights after 100s learning: The weight range is between $w_{min} = 0.001$ and $w_{max} = 2$
top: There is now more connectivity preserved and there are more different weight values.
bottom: The input and recurrent weights are similar distributed and both are mostly below 1.0.

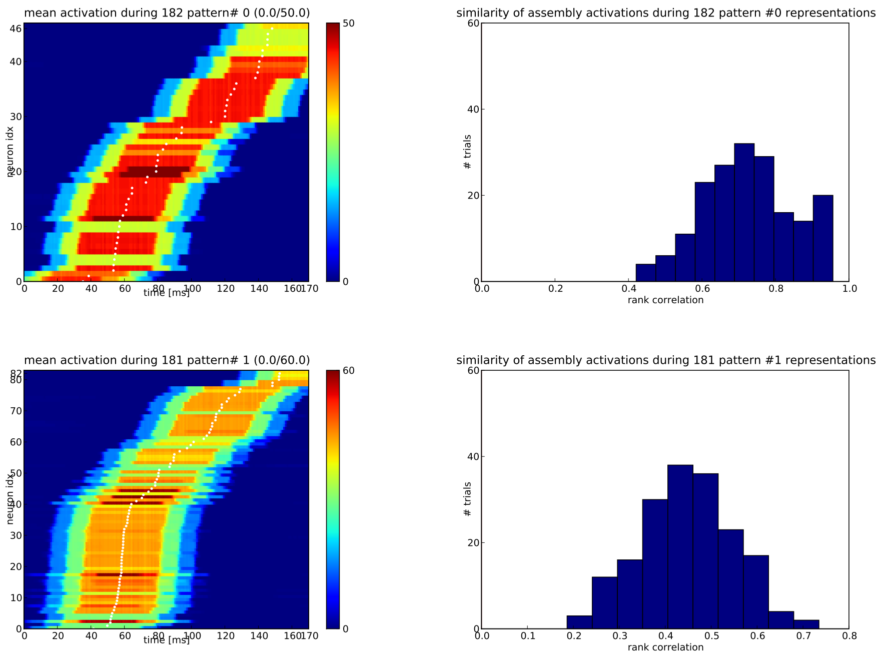
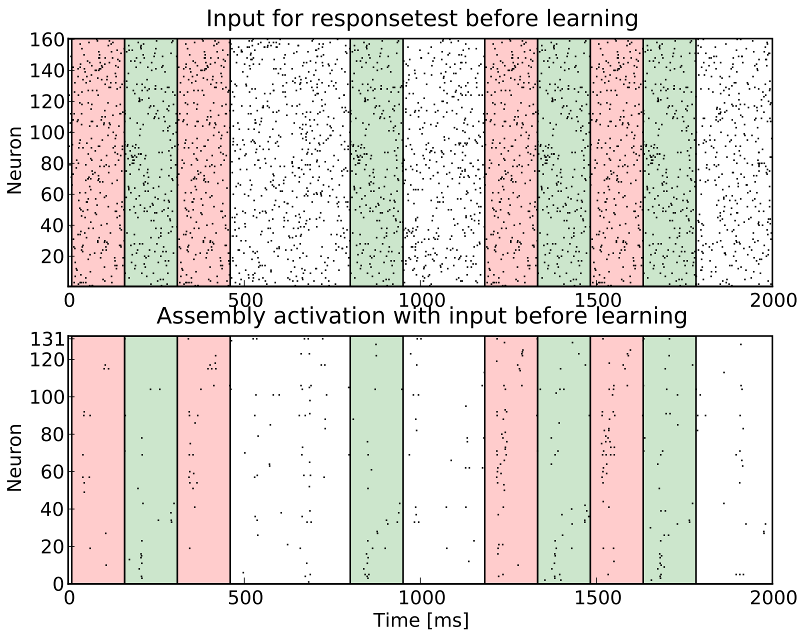


Figure 5.3: Setup 1 mean activation and rank order correlation for all assembly neurons with a maximum rate above 30Hz:
Assembly 1 first row: the assigned neurons have specialized clearly on specific time points within the presentation time of pattern 0. The rank order correlation is also stable.
Assembly 2 second row: also the second trajectory has a clear activation sequence and high rank order correlation.



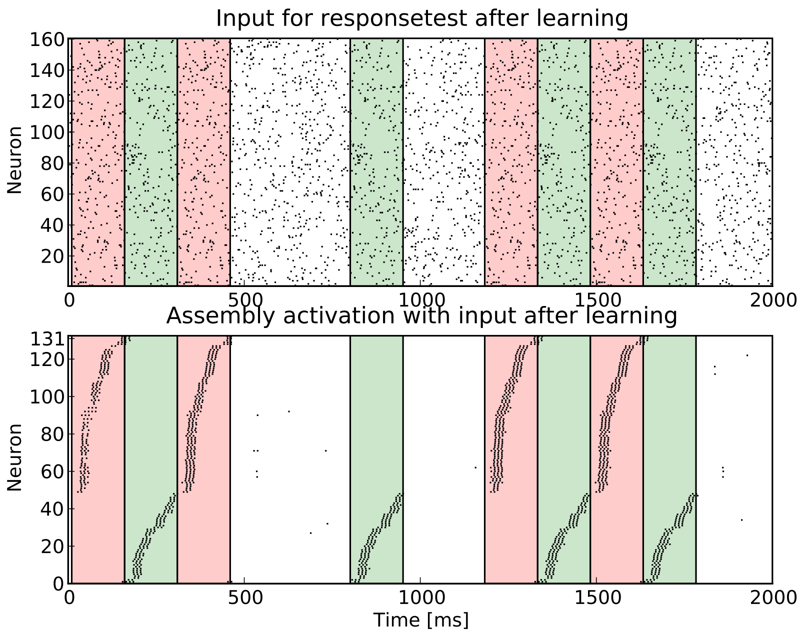


Figure 5.4: The exact same spike time pattern is shown to the untrained network (top) and the trained network (bottom). The neurons are ordered by their mean activation time for the corresponding assembly, overlapping neurons are plotted twice according to their order. **top:** We see that there is no structured activity, each response to a pattern is completely different. **bottom:** Each neuron has specialized for a specific time point when the stimulus is shown.

5.2 Setup 2: Internal activity

The initial setup becomes very input driven because all weights are reduced to a low value in a very short time. With setup 2 we are trying to force the network to use more recurrent connections by decoupling some SWTAs from the external input. To generate internal activity we need some stimulation for these SWTAs. Therefore channels that deliver only Poisson noise without any pattern instead of the external channels, are connected with a probability of 0.3.

5.2.1 Learn 2 patterns with setup 2

The randomly drawn structure is plotted in figure 5.5. It almost looks like setup 0, because the weights are randomly initialized. But only 4 SWTAs (44.4%) get a real signal, 5 SWTAs (65.6%) receive their channels just Poisson spiketrains at the same rate without a pattern. The SWTA 1, 2, 4, 5 and 6, which receive just noise are clearly visible after learning in figure 5.6, because they have more uniform weight values.

Both assemblies are spread over all 9 SWTAs and there is a overlap of 11 neurons between them. So each non-input SWTAs takes part by both assembly activation trajectories. This indicates that the recurrent connections are used for learning the activation trajectories.
The assembly of figure 5.7 has some similarity to figure 4.11 from setup 0. But in this configuration the assembly size with 44 and 45 neurons is more equal than the one in setup 121 to 90. The rank order correlation is also much higher, maybe because only neurons that highly correlate with the stimulus stay within the assembly.
A possible interpretation is, that the noise let the network forget weak ties of neurons to an assembly. This is helpful to preserve learning capability, so that the network is able to learn more at a later time. The presented simulation has 111 (49.6%) free neurons. The initial setup had none (0%) and setup 1 had 68 (28.2%).
There are just 4 SWTAs (44.4%) receiving the patterns and they get 64 channels which is a mean of 34.7% of their stimulation. So effectively, we have reduced the external stimulation by the pattern to around 15.4%. Sur-

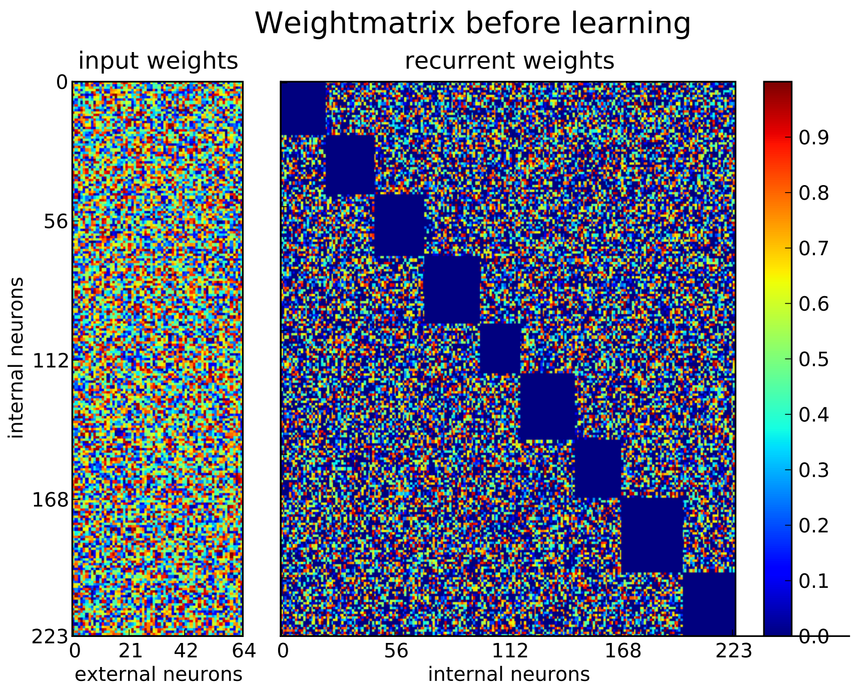


Figure 5.5: **Setup 2**: horizontal axis are the presynaptic neurons and vertically are the postsynaptic neurons: **input weights**: there are 64 external channels, each channel connects to all SWTA like in setup 0, which are 14336 input connections in total. So each SWTA receives a mean of 34.7% external input stimulation. But only 4 SWTAs (44.4%) get a real signal, 5 SWTAs (65.6%) receive their channels just noise at the same rate without a pattern. **recurrent weights**: SWTAs are all-to-all connected, each neuron connects with each other neuron with a probability of 0.6. This results in 26966 recurrent connections. The mean is 120.4 (min=102, max=139) connections, which is 65.3% of the stimulation. Each recurrent and input weight is initialized uniformly between 0.001 and 1.

prisingly this is enough to produce the same drawbacks as setup 0 (see section 4.4.2). So this structure inherits the problems from setup 0 even with this little input fraction. But the tendency to burst is not that strong, because the weights are smaller and the connectivity becomes very different. Nevertheless this setup offers some useful features.

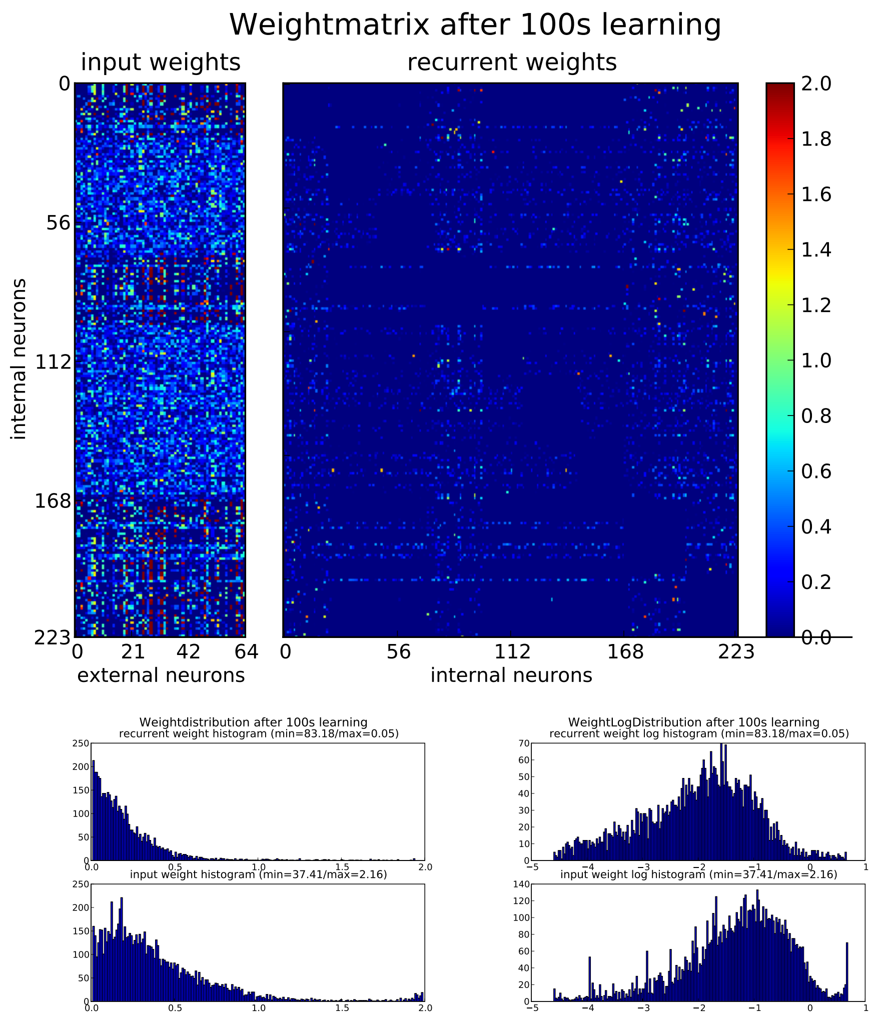


Figure 5.6: Synaptic weights of setup 2 after 100s learning. **external weights:** SWTA 1, 2, 4, 5 and 6 have no external connectivity, their weights are more uniform as they just receive random Poisson spike trains. SWTA 0, 3, 7 and 8 have a similar connectivity than in setup 0, but now the most input weights are below 1, which prevents the bursting tendency. **internal weights:** all SWTAs are part of the both assemblies, this indicates that the recurrent structure is used for learning. The most recurrent weight values are below 0.6.

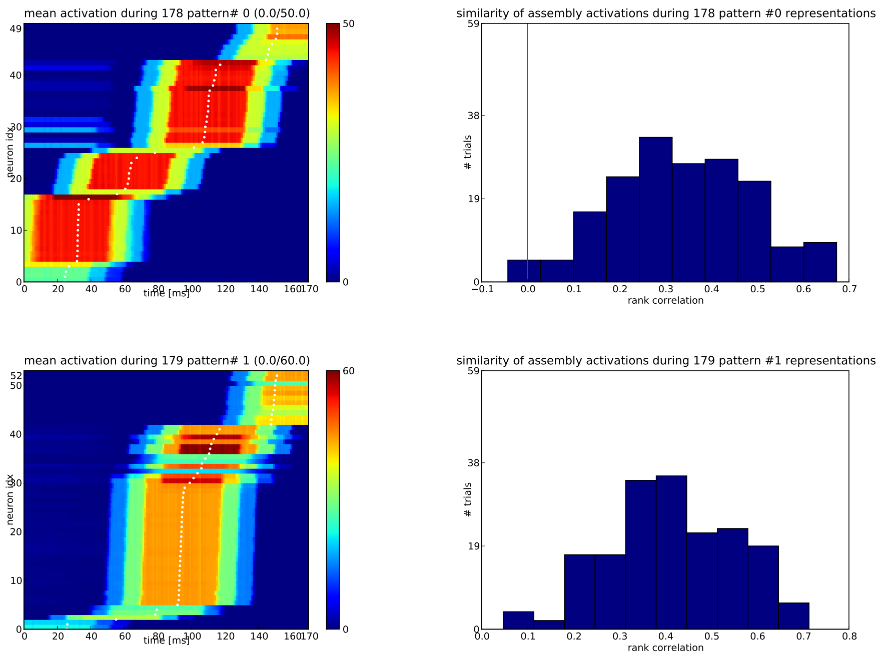


Figure 5.7: Mean activation and rank order correlation of neurons with a maximum rate above $20Hz$: The first assembly is spread over the whole presentation time and the second has a big gap at the beginning. Many neurons are specialized for a similar specific time point, but fewer as in setup 0, so bursting is not a problem in this setup. The activation trajectory is quite stable, as the rank correlation is positive.

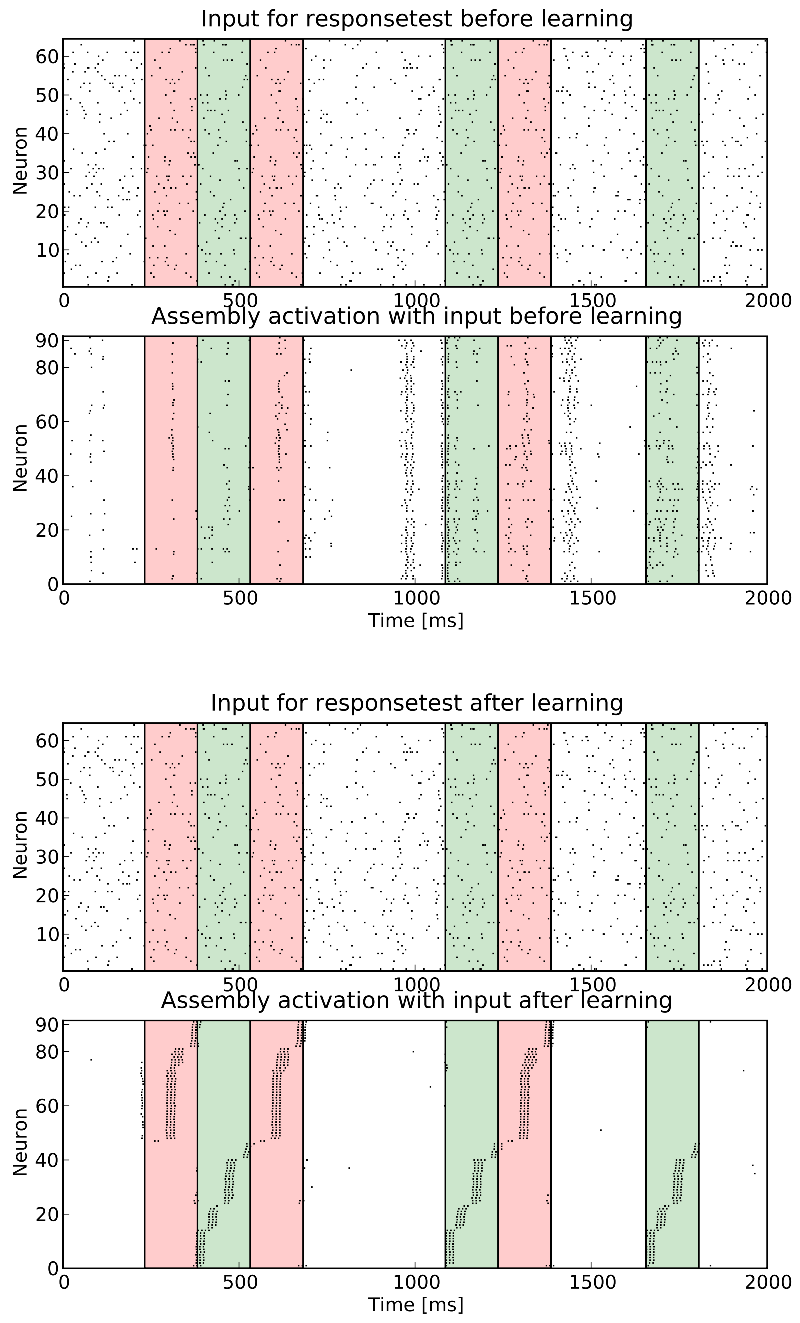


Figure 5.8: The exact same spike time pattern is shown to the untrained network (top) and the trained network (bottom). The neurons are ordered by their mean activation time for the corresponding assembly, overlapping neurons are plotted twice according to their order. **top:** There is the tendency that the neurons fire together at random times, but there is no structured activity before learning. **bottom:** A stable activation trajectory for both patterns is visible, but there is a gab in the middle of the first train and at the beginning of the second.

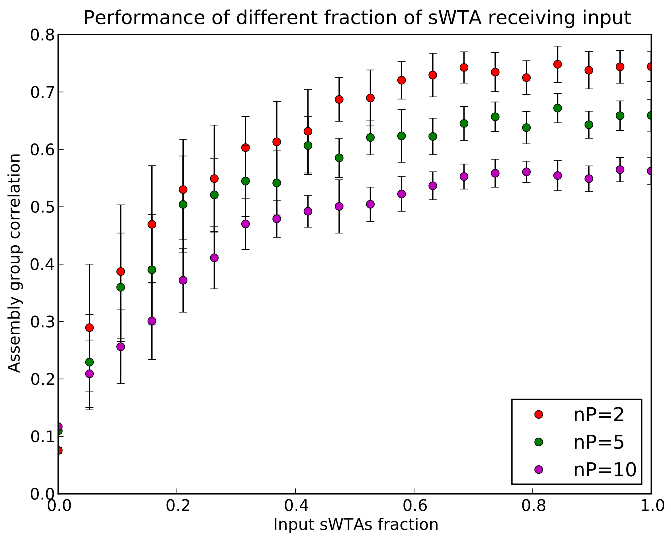


Figure 5.9: Performance of different input fractions to learn 2, 5 and 10 patterns over 50 trials with standard error.

5.3 Setup 3: Different external input with internal activity

The previous setup had some useful features, but the same drawback as the initial setup. So we combine these features with the good learning capability of setup 1.
At first the new parameter for the input fraction of external input receiving SWTAs is further analyzed. According to figure 5.9, even with an input fraction of 0.35 emergence of good input specific assemblies is possible and above 0.60 a good performance is always achieved. We will present now an example with an input fraction of 0.5.

5.3.1 Learn 2 patterns with setup 3

The structure in figure 5.10 is now a combination of setup 1 and 2. SWTA 3, 4, 7 and 8 are connected to the external channels which receive input. All other receive just Poisson spike trains without patterns. The synaptic weights after 100s learning in figure 5.11 show a stronger connectivity than all previous analyzed structures. We have very few weights (1.7%) at

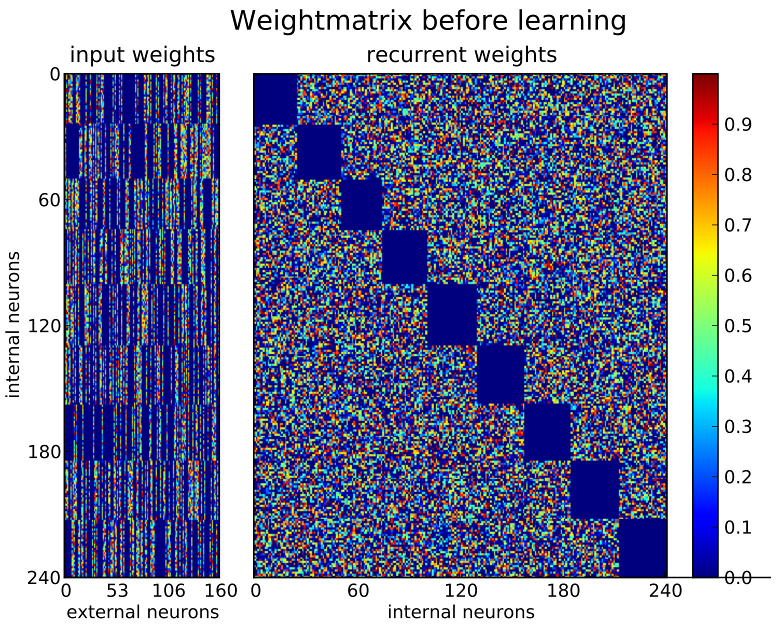


Figure 5.10: **Setup 3**: horizontal axis are the presynaptic neurons and vertically are the postsynaptic neurons: **input weights**: there are 160 external channels, each channel connects to a SWTA with probability 0.4, this results in a mean of 62 (min=44, max=73) connections per neuron. In total there are 31177 recurrent connections and 14954 input connections. So each SWTA receives a mean of 32.3% external input stimulation. But only 4 SWTAs (44.4%) get a real signal, 5 SWTAs (65.6%) receive their channels just noise at the same rate without a pattern. So only 14.3% of the channels receive the embedded pattern. **recurrent weights**: SWTAs are all-to-all connected, each neuron connects with each other neuron with a probability of 0.6, which results in a mean of 129 (min=110, max=144) connections, which is 67.3% of the stimulation. Each recurrent and input weight is initialized uniformly between 0.001 and 1.

maximum $w_{max} = 2$ and fewer (28%) at minimum $w_{min} = 0.001$. Now the input and recurrent weights are similar distributed, it has a similarity to a log-normal Distribution.

The first assembly has a correlation of 0.84 with 26 neurons (10.8%), the second has a correlation of 0.73 and 35 neurons (14.5%). But both assemblies have only neurons in the input receiving SWTAs 3, 4, 7 and 8. Nevertheless there are 3 overlapping neurons. This setup is capable to create an efficient input specific assembly trajectory for an input and preserve also capabilities for feature stimuli.

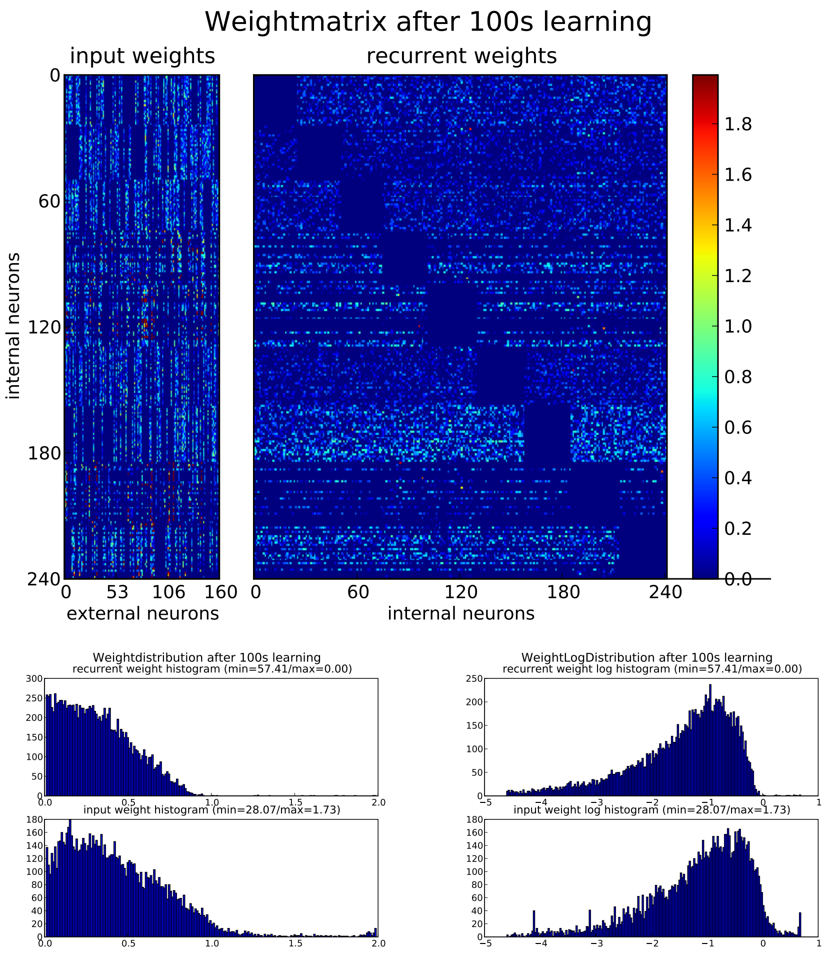


Figure 5.11: Setup 3 synaptic weights after 100s learning: There is a strong connectivity with nearly no weights at maximum and fewer at minimum. The distribution for input and recurrent weights is quite similar and resembles a log normal distribution.

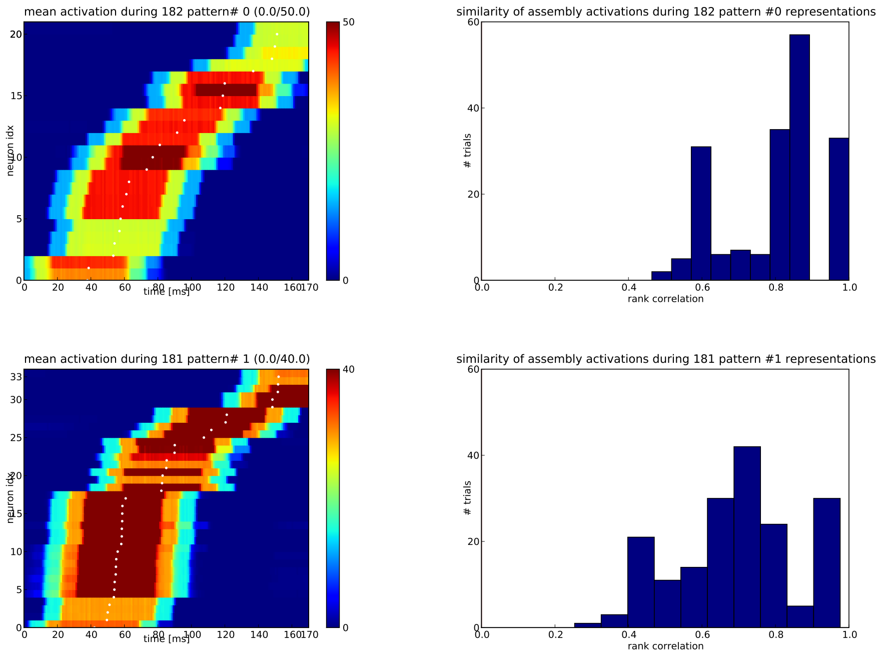


Figure 5.12: Setup 3 mean activation and rank order correlation for neurons with maximum rate above $30Hz$: The assemblies are smaller, but there are neurons specialized to all time points. The rank order is very stable, as the rank correlation is very high and the neurons fire at specific time points with high rate.

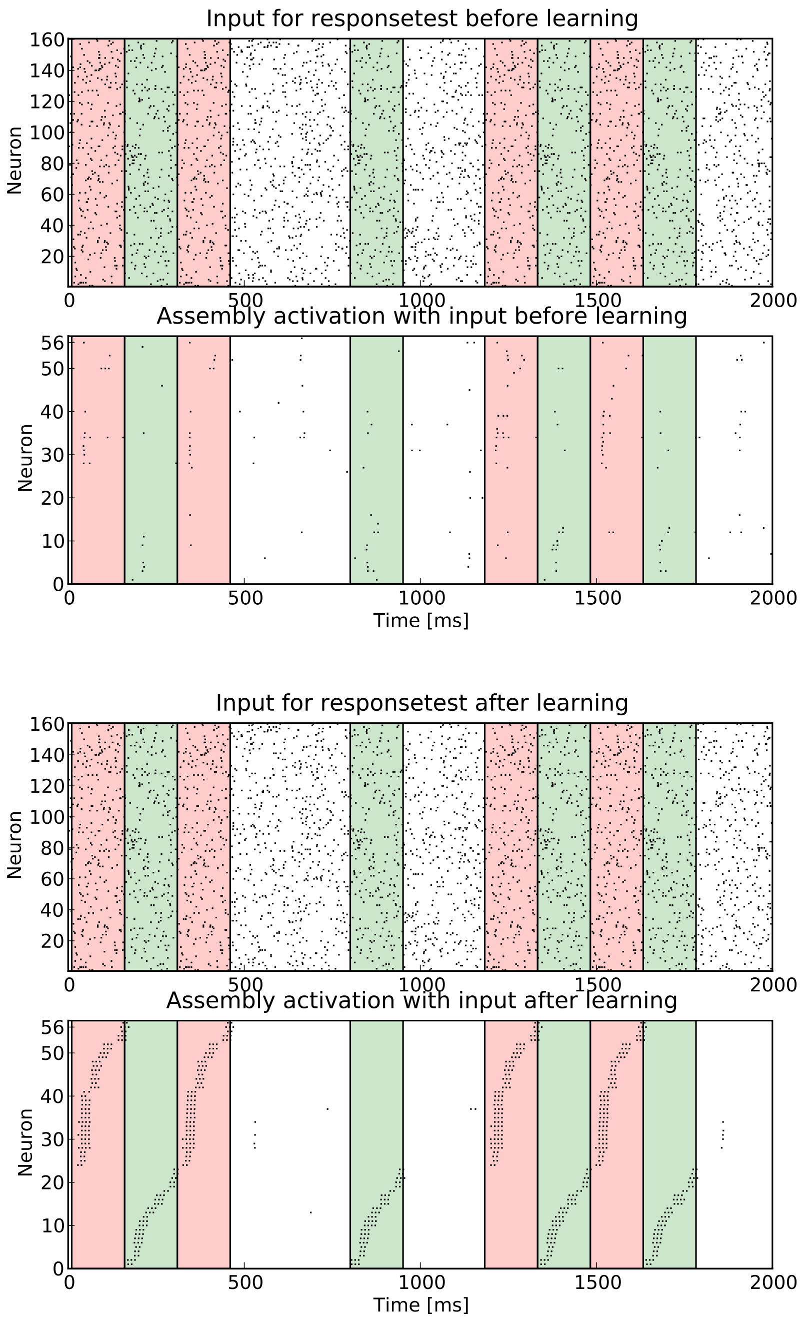


Figure 5.13: The exact same spike time pattern is shown to the untrained network (top) and the trained network (bottom). The neurons are ordered by their mean activation time for the corresponding assembly, overlapping neurons are plotted twice according to their order. **top:** There is no structured activity. **bottom:** There is a stable activation trajectory for both patterns visible, which resemble each other. Between the representation times there are nearly no spikes.

5.4 Noise as feature

One side effect of these structures is that they induce some amount of noise, which has some functions in matter of varying the stimuli and making the learning more robust. In this section we analyze how much additional noise could be handled by setup 1.

5.4.1 Superimposed noise

There is already some significant amount of noise within the signal. We embed the pattern within the stimulus, so we are switching between 100% noise and pure signal. In this section we put some additional noise on top of the pattern. Therefore there will be noise in all channels all the time.
In figure 5.14 on the left side we are putting Poisson spike trains as noise on top of a $r_{in} = 8Hz$ signal. Therefore a noise rate of $r_{Noise} = 4Hz$, means that there is a constant rate of $r_{total} = 12Hz$.
Surprisingly using additional noise on top improves the learning performance. A total rate of $r_{total} = 10Hz$ worked best. This could indicate that the presented stimulation r_{total} is more important then the pattern rate r_{in}.
To test this assumption we let the total stimulation r_{total} constant and reduce the rate of the signal exactly by the rate of the noise. The result is plotted in figure 5.14 on the right side.
We see that 10 patterns could be learned even with 30% noise. Using even 50% noise works for 2 signals, but 60% seems to be the borderline.

This setup is capable to successfully build self organized assemblies of neurons even when the pattern is embedded in huge amount of noise.

5.4.2 Jitter

Another possible noise source is random time delays of transmission, which result in slightly different spike times. This could be modeled by using a noise on the time scale while presenting the spike time pattern. This noise was a Gaussian on the spike times, which mean is the time point of a spike and a variance of the jitter-value in ms. In figure 5.15 a small

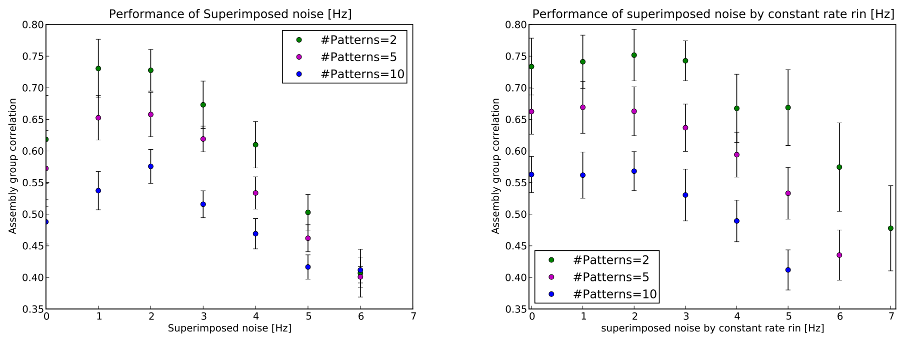


Figure 5.14: Performance of superimposed noise on the frozen pattern with mean performance and standard error of 50 trials. **left:** Superimposed noise on top of $r_{in} = 8Hz$ signal. Adding $r_{Noise} = 2Hz$ on top has even a better performance. **right:** The total rate is constant $r_{total} = 10Hz$. The performance is also slightly better with 2 Hz Noise, drops at $r_{Noise} = 5Hz$ which is 50% noise during pattern representation times.

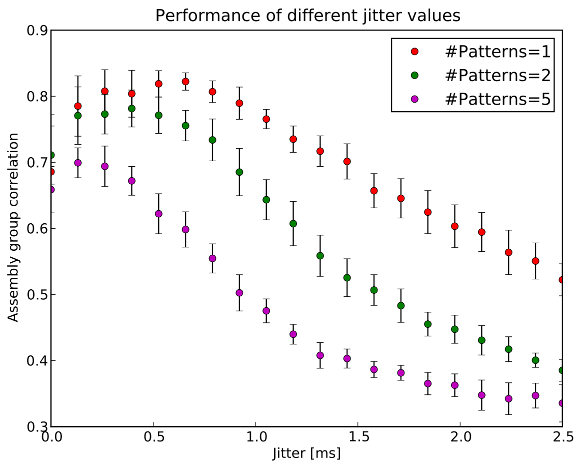


Figure 5.15: Performance of different jitter values from 100 trials with standard error. A jitter value of $0.8ms$ works best for 1 and 2 patterns. By using 5 pattern the performance is best with $0.15ms$. This could be due to a limited network capacity, as jittering could be seen as showing more patterns to the network.

jitter on the time axis has a positive influence on the learning performance. Slight jittering could have improve the learning performance in showing more variations of the pattern to the network, which could makes the learned representation more robust. But this reduces the performance for learning more patterns, as jittered patterns could need more network capacity as it could be seen as more different patterns.

5.4.3 Synaptic delays

Different transmission times of synapses are modeled as delays. We used $3ms$ for the excitatory synapses and $2ms$ for the inhibitory synapses. This times are randomly drawn for each synapse according to a Gaussian Distribution with mean of the delay time and variance of half the delay time.
This was necessary to make bursting less probable, as there was a problem that closely spiking neurons of different SWTAs started to synchronize their firing behavior and so facilitated bursting. Using different delay times improved the learning performance.
A possible explanation is, that the same external channel produces different internal activity. Somehow this could be compared to the jittering effect, as different forms of the pattern get in the network. But the main difference is, that the amount of pattern variation is fixed through the structure and therefore there is no capacity problem. Therefore all simulations use random synaptic delays at each synapse.

Chapter 6

Biologically inspired setup

In this chapter we will show that the determined setup could be scaled up to much bigger configurations and is also able to learn efficiently by incorporating various different biologically inspired features.
At first we will introduce distance dependent connectivity. Afterwards we will add parts from the previous chapters to the setup. The final configuration will be setup 3 with superimposed noise on top of the pattern and also noise on the time axis on a much bigger $10x10$ grid of 100 SWTAs with an exponentially distributed distance dependent connectivity pattern.

6.1 Distance dependent connectivity

6.1.1 Definition

To build a biologically more realistic setup like [Markov et al., 2011] we use a distance dependent connectivity pattern according to an exponential distribution. So the probability that two neurons are connected drops exponentially with the distance. Now we calculate all distances between the SWTAs on the 2D grid and draw for each neuron according to the exponential distribution from equation 6.1.

$$p(d) = \lambda exp(-\lambda d) \tag{6.1}$$

6.1.2 Periodic boundary conditions

Because we wanted to exclude border effects of small setups, we added a periodic boundary condition. By calculating the distance the whole grid is repeated 9 times (like a $3x3$ grid) and the starting point is the location in the center grid and the end point is the location on the closed grid. Therefore the resulting distances are a little shorter, but each point get a similar amount of distant neurons, which emulates a much bigger grid. The synaptic distribution histogram using $\lambda = 0.455$ is plotted in figure 6.2.

6.1.3 Performance

To show how the exponentially distributed connectivity influences the performance, we use 50 SWTAs on a $5x10$ grid, with the same configuration as in the previous chapters and evaluate the performance by learning 2, 5 and 10 input patterns.
The results are in figure 6.1. For 2 patterns a higher λ does not drop the performance. Surprisingly very few connections with $\lambda = 0.1$ and many $\lambda = 1.1$ works best for 5 and 10 patterns. The performance of 5 patterns is mainly constant between 0.2 and 1 at 0.59.

6.2 Biologically inspired setup simulation

We are using now setup 3 with a $10x10$ grid of 100 SWTAs of 2692 neurons. The hidden neurons are randomly drawn within a range of $[24, 30]$. The neurons are connected according to an exponential distance distribution with $\lambda = 0.455$, which results in 1422294 recurrent connections. The histogram of distances is plotted in figure 6.2. The input rate $r_{in} = 8Hz$ with superimposed $2Hz$ Poisson spike train as noise.
The external input fraction is only 10.7% of the connection per neuron. There are only 48 SWTAs connected to the real input and 52 receive just noise. Only a fraction of 5.14 % of the input channels carry the spike time pattern. There is much activity within the network, as the neurons are spiking because of the Poisson spike train.

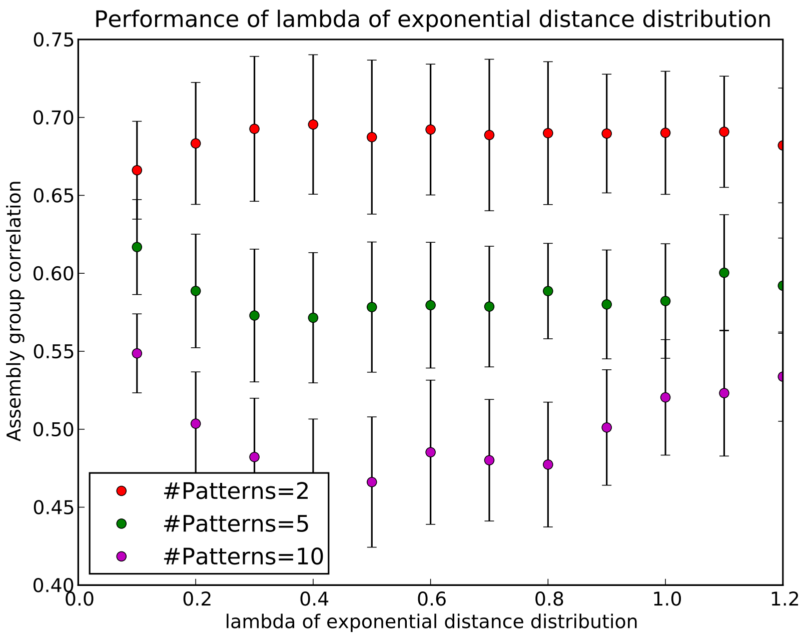


Figure 6.1: Performance of different λ of the exponential distance distribution connectivity probability over 60 trials using setup 3 with 0.5 input fraction.

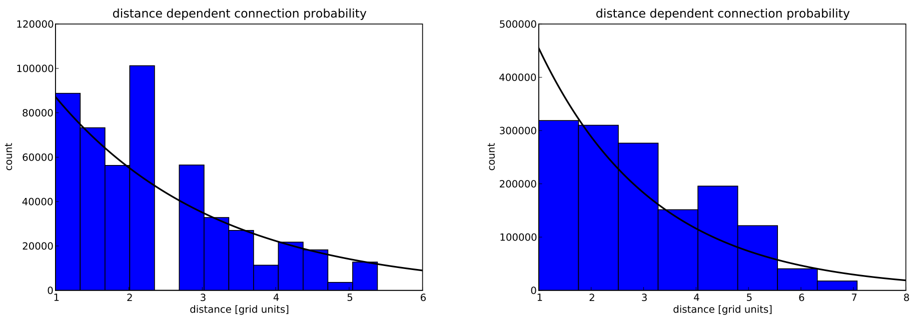


Figure 6.2: Histogram of distances with $\lambda = 0.455$: **left:** The below presented setup with 504353 recurrent connections, which is used for the performance test. The gap by 2.5 is due to the periodic boundary conditions, as the lower border is just 5. This vanish in the right setup. **right**: The histogram of the 100 SWTAs on a $10x10$ grid with 1422294 recurrent connections, which is simulated in the next chapter.

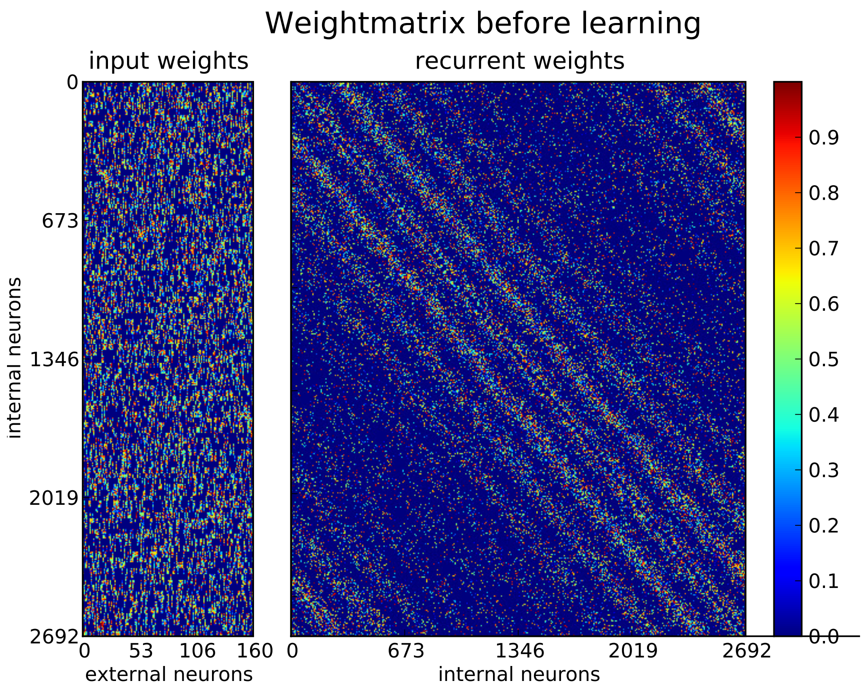


Figure 6.3: Distance dependent connectivity initial synaptic weights: horizontal axis are the presynaptic neurons and vertically are the postsynaptic neurons. Each recurrent and input weight is initialized uniformly between $w_{min} = 0.001$ and 1. **input weights**: there are 160 external channels, each channel connects to a SWTA with probability 0.4, this results in a mean of 64 (min=46, max=77) connections per neuron. In total there are 171681 input connections. So each SWTA receives a mean of 10.7% external input channels, whereas only 48 (48%) of them receive the pattern. Therefore only 5.14% of the channels receive input. **recurrent weights**: the recurrent connection is drawn for each neuron according to the exponential distance distribution, this results in 1422294 recurrent connection, which is a mean of 528 connections per neuron (min=446, max=592)

Result after 100s learning

The first assembly has a correlation of 0.74 with 195 neurons which is 7.2% of the network, which spreads over 54 SWTAs. The second assembly has a correlation of 0.76 with 299 neurons which is 11.1%, which spreads over 85 SWTAs. There are only 2 Neurons that are part of both assemblies.

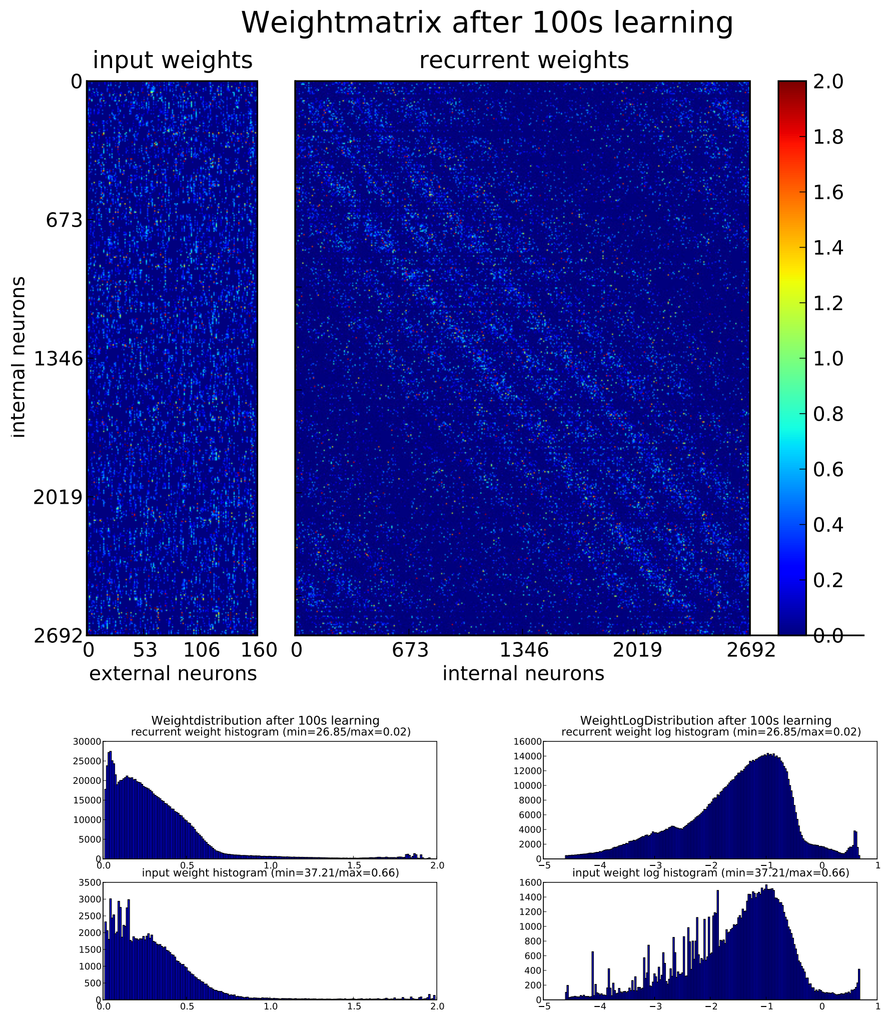


Figure 6.4: Synaptic weights after 100s learning:
top: The connections are mainly preserved, the diagonals of higher connectivity are still visible.
bottom: The histogram shows a similar distribution for recurrent and input weights. There are very few weights at maximum (0.02% recurrent and 0.66% input) and only 26.9% recurrent and 37.2% input weights go down to w_{min}.

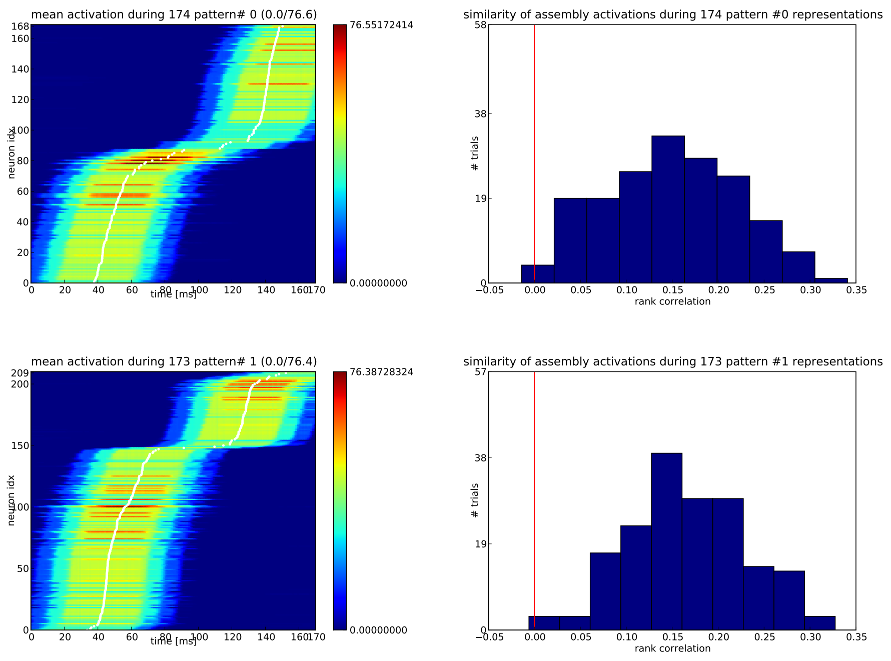


Figure 6.5: Mean activation and rank correlation for neurons with maximum rate above $30Hz$: There is a clear activation trajectory and the rank order correlation is positive

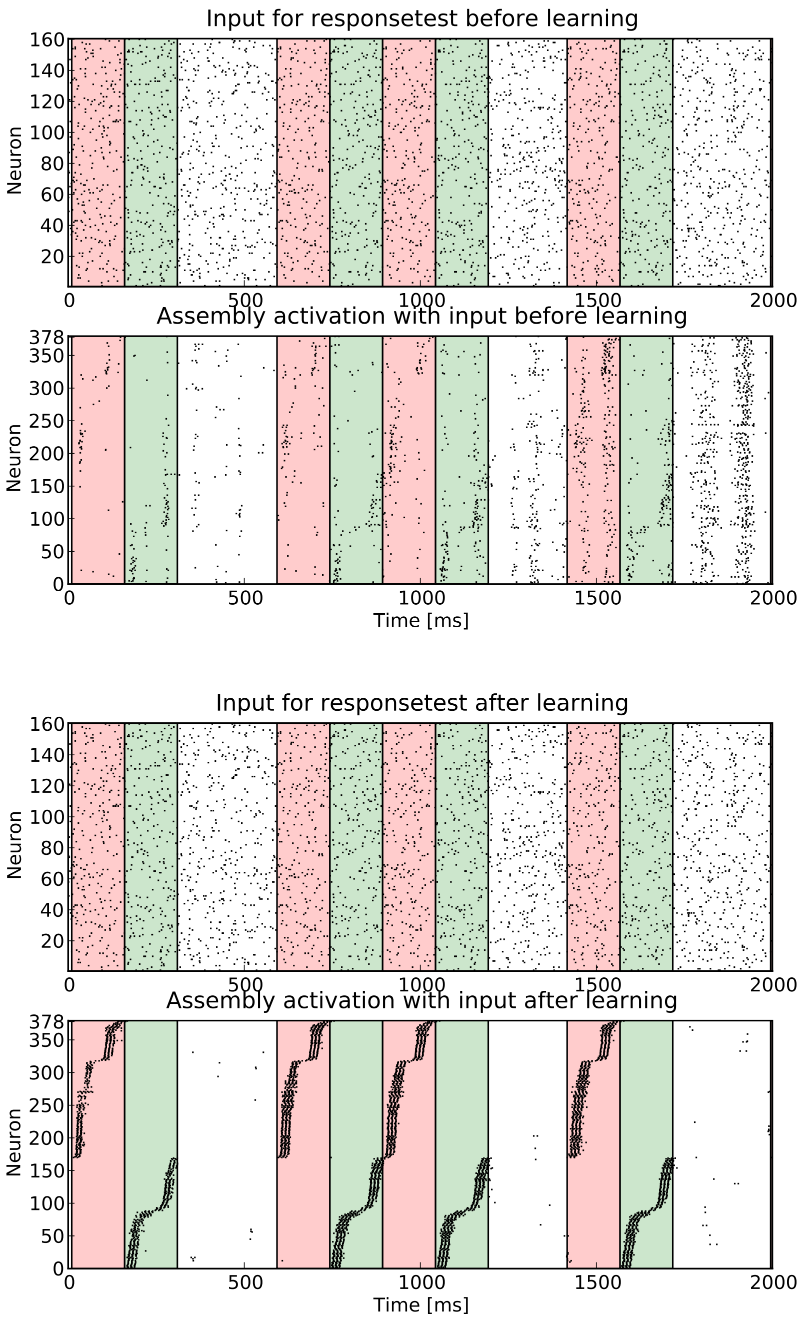


Figure 6.6: The exact same spike time pattern is shown to the untrained network (top) and the trained network (bottom). The neurons are ordered by their mean activation time for the corresponding assembly, overlapping neurons are plotted twice according to their order: **top:** There is again no structured activity before learning. **bottom:** There is clear activation trajectory for each pattern.

Chapter 7

Summary and conclusion

In this thesis we studied the self organization of neurons in different network structures. Our goal was to find and describe networks that are capable to emerge stimulus specific assemblies and have various biological properties.

In order to do that, a simulation environment was developed and used to simulate various different configurations. We have presented the basic micro-circuit motif, how these models are implemented and recurrently connected. Afterwards we have defined a performance criterion, that was applied to quantify the ability of networks to emerge assemblies. This criterion was used to evaluate different configurations and describe how variations of different parameters influence the self organization capability of the network. We have described and solved several difficult issues in constructing a stable configuration. In chapter 5 we have analyzed different structures with its specific features and learning capabilities. Afterwards different kinds of noise were examined. It was shown that the networks could handle big amount of noise, if the network stimulation is within a working regime.
The final simulation uses 100 sWTAs in a distance dependent connectivity pattern with the previously defined model parameters. In this simulation only 5% of the channels receive an external pattern with additional superimposed noise at random times.

This thesis has shown how to construct a scalable recurrent SWTA circuit that is capable to emerge input specific assemblies, even with very little input stimulation and high amount of noise. Furthermore neurons are part of different assembly groups and spike less stereotypical, which resembles some data of [Harvey et al., 2012]. So maybe the presented results could contribute to the understanding and modeling of biological data.

Future research could go in various different directions:

- As the presented networks could mimic various biological features and is scalable, it could maybe be used to model larger neural structures. One approach would be to use different SWTA configurations within one recurrent setup. Then different SWTAs would respond to different input complexities, which could enhance the network to work for various different amount of input stimulation and so extend the working regime.
- Another approach is to enhance the model itself. This was mentioned in section 4.1.3. Then each SWTA would be capable to adapt it's excitability α and specialize on a specific input complexity.
- Another possibility is to go one step further and start chaining different learned assemblies. The input patterns could for example be different syllables. The network could be trained to construct different assemblies to each one. Afterwards the chaining of this assemblies and there behavior could be further analyzed and compared to biological data.

Bibliography

[Büsing et al., 2011] Büsing, L., Bill, J., Nessler, B., and Maass, W. (2011). Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons. *PLoS Computational Biology*, 7(11):e1002211. (Cited on pages 6 and 16.)

[Buzsáki, 2010] Buzsáki, G. (2010). Neural syntax: cell assemblies, synapsembles, and readers. *Neuron*, 68(3):362–385. (Cited on page 14.)

[Habenschuss et al., 2013] Habenschuss, S., Puhr, H., and Maass, W. (2013). Emergence of optimal decoding of population codes through STDP. *Neural Computation*, 25(6):1371–1407. (Cited on page 22.)

[Harvey et al., 2012] Harvey, C. D., Coen, P., and Tank, D. W. (2012). Choice-specific sequences in parietal cortex during a virtual-navigation decision task. *Nature*, 484(7392):62–68. (Cited on pages 14 and 69.)

[Hebb, 1949] Hebb, D. O. (1949). *The organization of behavior: A neuropsychological approach.* John Wiley & Sons. (Cited on pages 1 and 14.)

[Hunter, 2007] Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95. (Cited on page 7.)

[Jolivet et al., 2006] Jolivet, R., Rauch, A., Lüscher, H.-R., and Gerstner, W. (2006). Predicting spike timing of neocortical pyramidal neurons by simple threshold models. *Journal of computational neuroscience*, 21(1):35–49. (Cited on page 8.)

[Jones et al., 01] Jones, E., Oliphant, T., Peterson, P., et al. (2001–). SciPy: Open source scientific tools for Python. (Cited on page 7.)

[Jonke, 2011] Jonke, Z. (2011). Technical report: swta. SWTA Report. (Cited on page 5.)

[Jonke et al., 2013] Jonke, Z., Legenstein, R., Habenschuss, S., and Maass, W. (2013). Emergent coding of features and bayesian computation in ensembles of pyramidal cell with relaxed lateral inhibition. (Cited on pages 1, 4 and 7.)

[Keck et al., 2012] Keck, C., Savin, C., and Lücke, J. (2012). Feedforward inhibition and synaptic scaling–two sides of the same coin? *PLoS computational biology*, 8(3):e1002432. (Cited on page 22.)

[Klampfl and Maass, 2013] Klampfl, S. and Maass, W. (2013). Emergence of dynamic memory traces in cortical microcircuit models through STDP. *The Journal of Neuroscience*, 33(28):11515–11529. (Cited on pages 1 and 2.)

[Luczak et al., 2009] Luczak, A., Barthó, P., and Harris, K. D. (2009). Spontaneous events outline the realm of possible sensory responses in neocortical populations. *Neuron*, 62(3):413–425. (Cited on page 18.)

[Markov et al., 2011] Markov, N., Misery, P., Falchier, A., Lamy, C., Vezoli, J., Quilodran, R., Gariel, M., Giroud, P., Ercsey-Ravasz, M., Pilaz, L., et al. (2011). Weight consistency specifies regularities of macaque cortical networks. *Cerebral Cortex*, 21(6):1254–1272. (Cited on page 60.)

[Markram et al., 1998] Markram, H., Wang, Y., and Tsodyks, M. (1998). Differential signaling via the same axon of neocortical pyramidal neurons. *Proceedings of the National Academy of Sciences*, 95(9):5323–5328. (Cited on page 13.)

[Nessler et al., 2013] Nessler, B., Pfeiffer, M., Buesing, L., and Maass, W. (2013). Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity. *PLOS Computational Biology*, 9(4):e1003037. (Cited on page 1.)

[Pecevski et al., 2009] Pecevski, D., Natschläger, T., and Schuch, K. (2009). Pcsim: a parallel simulation environment for neural circuits fully integrated with python. *Frontiers in neuroinformatics*, 3. (Cited on page 7.)

[Zeno, 2012] Zeno, J. (2012). Report: K-wta (no.5.02). (Cited on page 22.)