Int. J. Appl. Math. Comput. Sci., 2008, Vol. 18, No. 2, 117–127
DOI: 10.2478/v10006-008-0011-1

amcs

ANALYSIS OF THE RESUME LEARNING PROCESS FOR SPIKING NEURAL NETWORKS

FILIP PONULAK

Institute of Control and Information Engineering
Poznań University of Technology,
ul. Piotrowo 3a, 60–965 Poznań, Poland
e-mail:Filip.Ponulak@put.poznan.pl

In this paper we perform an analysis of the learning process with the ReSuMe method and spiking neural networks (Ponulak, 2005; Ponulak, 2006b). We investigate how the particular parameters of the learning algorithm affect the process of learning. We consider the issue of speeding up the adaptation process, while maintaining the stability of the optimal solution. This is an important issue in many real-life tasks where the neural networks are applied and where the fast learning convergence is highly desirable.

Keywords: supervised learning, spiking neural networks, parametric analysis, learning window.

1. Introduction

A rapidly growing interest in applications of Artificial Neural Networks (ANNs) in real-life systems has been observed in the last decades (Freeman and Skapura, 1991; Korbicz, Obuchowicz and Uciński, 1994; Kangas and Kohonen, 1996; Papik, Molnar, Schaefer, Dombovari, Tulassay and Feher, 1998; **?**). The fundamental property of ANNs spurring this interest is their learning ability. ANNs can adapt to changes in their environment. From the engineering point of view, it is desirable to reduce the time required for the adaptation process, while maintaining the stability of the optimal state of the network (Hertz, Krogh and Palmer, 1991). Here we consider this issue in the context of Spiking Neural Networks (SNNs) (Gerstner and Kistler, 2002b; Maass and Bishop, 1999) and a ReSuMe learning method (Ponulak, 2005; Kasiński and Ponulak, 2005; Ponulak, 2006b).

ReSuMe (or Remote Supervised Method) is a learning algorithm dedicated for spiking neurons. The method corresponds to the Widrow-Hoff rule, but also takes advantage of spike-based plasticity mechanisms such as Spike-Timing Dependent Plasticity (STDP) (Gerstner and Kistler, 2002a; Gerstner and Kistler, 2002b). It was shown that ReSuMe enables effective learning of complex temporal and spatio-temporal spike patterns with a given accuracy and that the method allows imposing on the networks the desired input/output behaviour (Kasiński and Ponulak, 2005). Generalization properties of spiking neurons trained with ReSuMe were demonstrated in (Ponulak and Kasiński, 2006a). Several applications of the ReSuMe method were discussed in (Ponulak and Kasiński, 2005; Ponulak and Kasiński, 2006b; Ponulak, Belter and Kasiński, 2006).

The goal of this study is to perform a thorough analysis of the ReSuMe learning process and to determine how the particular adjustable parameters of the ReSuMe algorithm influence the learning convergence. The obtained results will be evaluated in order to ascertain the optimal parameter values leading to the fastest learning convergence.

2. Learning algorithm and the investigation procedure

We consider the ReSuMe learning rule given by the following equation (Ponulak, 2006b):

$$\frac{\mathrm{d}}{\mathrm{d}t}w_{ki}(t)=\left[S^d(t)-S^o(t)\right]\left[a+\int_0^\infty W(s)\,S^{in}(t-s)\,\mathrm{d}s\right], \tag{1}$$

where $S^d(t)$, $S^{in}(t)$ and $S^o(t)$ are the desired, pre- and postsynaptic spike trains (Gerstner and Kistler, 2002b), respectively. The constant a represents the so-called non-Hebbian contribution to the weight changes. The function $W(s)$ of a time delay s between the correlated

spikes is known as a learning window (Gerstner and Kistler, 2002b). The shapes of $W(s)$ applied in ReSuMe are similar to the ones used in STDP models and can be mathematically described in the following form:

$$W(s) = \begin{cases} +A_+ \cdot \exp\left(-s/\tau_+\right) & \text{if} \quad s \geq 0, \\ -A_- \cdot \exp\left(s/\tau_-\right) & \text{if} \quad s < 0, \end{cases} \tag{2}$$

with amplitudes $A_+, A_- \geq 0$ and time constants $\tau_+, \tau_- > 0$ of the positive and negative parts of the learning window, respectively, see left-hand side panels in Fig. 2. In our analysis we consider the ReSuMe learning algorithm implemented in the LSM (Liquid State Machine) network architecture (Maass, Natschlaeger and Markram, 2002; Natschlaeger, Maass and Markram, 2002). The network consists of a large, recurrent neural structure NMC (Neural Microcircuit) and a set of readout neurons (the output layer). In our study the NMC, containing 1600 LIF (Leaky-Integrate and Fire, (Gerstner and Kistler, 2002b)) neurons, is driven with a single input signal. The outputs of individual NMC neurons are projected onto a single readout LIF-neuron, considered as the network's output and trained to reconstruct the predefined target spike train (for details on the methods and simulation parameters, see the Appendix).

Now, we investigate how the particular parameters $\boldsymbol{v} = \{a, A_+, A_-, \tau_+, \tau_-\}$ of the ReSuMe rule, Eqns. (1) and (2), influence the learning process.

The experimental procedure is summarized in Table 1. According to this procedure, in each experiment we modify a single, given parameter v_i. The range of the possible values $[v_i^{\min}, v_i^{\max}]$ of v_i is chosen to be wide enough to observe the various qualitative phenomena attributed to the learning process.

In order to obtain statistically reliable results, the implemented network is tested on 10 pairs (denoted by $\boldsymbol{T} = \{T_1, \ldots, T_{10}\}$) of the input and desired output signals applied to the network for every selected vector of the ReSuMe parameter values $\boldsymbol{v}$. The particular signals $\boldsymbol{T}$ are generated randomly according to the uniform distribution over a time period of 100 ms.

For each set $\boldsymbol{v}$ and T_j, $j = 1, \ldots, 10$, the learning process is repeated for 20 epochs. After every learning epoch m the performance index $P(m)$ is evaluated as a measure of the error between the desired $S^d(t)$ at the actual $S^o(t)$ network output (see Appendix for details). Finally, a sum $P_j = \sum_{m=3}^{20} P(m)$ is calculated over the last 18 epochs. The sum P_j can be considered as an approximation of the integral of $P(m)$ over m. P_j provides quantitative information on the quality of learning and the speed of convergence. Smaller values of P_j correspond to the better performance of the learning process. We observed that at the beginning of learning the value of the performance index $P(m)$ is much more influenced by the initial distance between the desired and actual output signals than by the learning process itself. For this reason we do not consider the first two epochs while computing P_j. The values of P_j calculated for each training pair T_j are collected in a set $\boldsymbol{P} = \{P_1, \ldots, P_{10}\}$, and the median $med(\boldsymbol{P})$ and standard deviation $std(\boldsymbol{P})$ of P are found.

The procedure described above is repeated for each value of v_i. Finally, $med(\boldsymbol{P})$ and $std(\boldsymbol{P})$ are plotted against the corresponding values of v_i.

The motivation for choosing the described investigation procedure is twofold. First, we observed that the influence of the particular parameters on the learning process is independent of the other parameters and so we are allowed to analyse each parameter separately. Second, the chosen procedure allows us to observe the new phenomena arising in the system under study whenever they occur. Since not all of these phenomena could be *a-priori* anticipated, some of them could be probably not accounted for by the optimization criteria if standard optimization techniques were used.

Based on the results of our previous experiments, we observed that satisfactory results of learning are obtained for the following set of parameter values: $a = 1 \times 10^{-2}$, $A_+ = 4 \times 10^{-10}$, $A_- = 1 \times 10^{-10}$, $\tau_+ = 2 \times 10^{-3}$, $\tau_- = 2 \times 10^{-3}$. These values are considered as our initial parameter settings and will be used in all experiments as default values.

3. Results

3.1. Non-Hebbian parameter (a). We begin with studying the influence of the non-Hebbian parameter a on the learning process. Suppose for a while that a is the main factor determining the weight change in (1). In such a case we can reduce the ReSuMe learning rule (1) to

$$\frac{\mathrm{d}}{\mathrm{d}t} w_{ki}(t) \cong \left[S^d(t) - S^o(t)\right] \cdot a. \tag{3}$$

According to this equation we note that for the case of excitatory synapses, where $a > 0$, the total weight change induced by the signals $S^o(t)$ and $S^d(t)$ is positive as long as the number of spikes in $S^d(t)$ is greater than in $S^o(t)$. This weight increase results in a stronger excitation of the postsynaptic neuron and, eventually, leads to its higher activity. On the other hand, the weight is decreased whenever the number of spikes in $S^o(t)$ is greater than desired. Opposite effects are observed for the case of inhibitory synapses and $a < 0$. There is no weight change if and only if the numbers of spikes in $S^o(t)$ and $S^d(t)$ are equal.

This analysis suggests that the role of a in the ReSuMe learning is to set the activity (the instantaneous firing rate) of the postsynaptic neuron to the level determined by the desired signal, but without taking into account a precise timing of particular spikes.

Table 1. Parametric analysis of the ReSuMe learning process. Algorithm of the experimental procedure.

Algorithm of the experimental procedure
Define a set of pairs of training signals: $\boldsymbol{T} = \{T_1, \ldots, T_{10}\}$
Define a set of ReSuMe parameters: $\boldsymbol{v} = \{v_1, \ldots, v_5\}$
For all predefined values of $v_i \in [v_i^{\min}, v_i^{\max}]$, with $i = 1, \ldots, 5$
For every $T_j \in \boldsymbol{T}$
Train the network with $\boldsymbol{v}$ and T_j for 20 learning epochs m
Compute $P_j = \sum_{m=3}^{20} P(m)$ for the performance indices $P(m)$
Collect a set $\boldsymbol{P} = \{P_1, \ldots, P_{10}\}$
Find median $med(\boldsymbol{P})$ and standard deviation $std(\boldsymbol{P})$
Plot $med(\boldsymbol{P})$ and $std(\boldsymbol{P})$ versus v_i

In the experiment considered the network is trained according to the ReSuMe learning rule given by (1), with a taking evenly spaced values from the range $[0, 0.1]$. The results demonstrate (Fig. 1(a)) that the median of the index $\boldsymbol{P}$ takes the lowest values, corresponding to the best performance of learning, for $a \in [0.01, 0.07]$. This suggests that too small or too high values of a are not optimal for learning. Indeed, if we analyse the performance index $P(m)$ calculated during the training with a taking the representative values $a = \{0, 0.025, 0.1, 0.2\}$, we observe the following (Fig. 1(b)):

- In the case of training with $a = 0$, i.e., with a below the optimal range, $P(m)$ decreases very slowly and reaches its minimal value, corresponding to obtaining a solution close to the desired one, not before the 20-th learning epoch.
- For $a = 0.1, 0.2$, a significant jump down of $P(m)$ occurs already after the first learning epoch. However, in the consecutive learning epochs $P(m)$ fluctuates around its minimal value, without settling there. This is a consequence of the too high weight changes forced by the high values of a.
- In the case of a taken from its optimal range (here $a = 0.025$), $P(m)$ decreases quickly enough and reaches its minimal level already after a few learning epochs. There exists an epoch m_s (here m_s=15), after which no fluctuations occur, and $P(m)$ remains close to zero, which confirms the stability of the obtained convergent process.

3.2. Amplitudes of the learning window (A_+, A_-). In the next experiment we consider the influence of the amplitudes of the learning window (Eqn. (2)) on the learning process. We analyse three cases with the amplitude $A_- = \{0.1A_+,\ 0.5A_+,\ A_+\}$ (Figs. 2 (a), (b) and (c), respectively). We tested the learning performance for A_+ taking values from the range $[10^{-11},\ 10^{-8}]$. In all three cases we observe (right-hand side panels in Fig. 2) that the index P has the minimum for the values of A_+ around $(1-3) \times 10^{-10}$, which suggests that for a given set of simulation parameters this range of values of A_+ is optimal. This demonstrates that too low or too high values of the learning rate (A_+,A_-) are not favourable for the good performance of the learning process.

By comparing the diagrams of $P(A_+)$ obtained for the three cases considered, we note that, generally, the best learning performance is observed for the case where $A_- = 0.1A_+$ and the worst one is recorded for $A_- = A_+$. This suggests that the influence of the negative part of the learning windows on the learning process is disadvantageous and that A_- should be significantly reduced or even rejected in order to improve the learning performance.

3.3. Time-constants of the learning window (τ_+, τ_-). The influence of the time-constants of the learning windows (τ_+, τ_-) on the learning performance is considered under several scenarios.

First, we assume that $\tau_- = \tau_+$ and that both time constants vary in a range of $[0.1,\ 10] \times 10^{-3}$ s. Under these conditions the best learning performance is obtained for $\tau_-, \tau_+ = [0.6,\ 1.5] \times 10^{-3}$ s (Fig. 3(b)).

From the definition of exponential learning windows (Eqn. (2)) it turns out that the effective time range of the learning window can be considered as $[-4\tau_-, 4\tau_+]$[1]. According to this remark, we note that the optimal values of the time constants $\tau_-, \tau_+ = (0.6 - 1.5) \times 10^{-3}$s imply the effective range of the learning window equal to $\pm(2.4 - 6) \times 10^{-3}$ s.

[1] For all arguments $s \geq 4\tau_+$ ($s \leq -4\tau_-$) the value of the learning window $W(s) \leq 0.05 \cdot A_+$ ($|W(s)| \leq |0.05 \cdot A_-|$). In such a case the influence of the Hebbian factor on the weight changes is negligible.

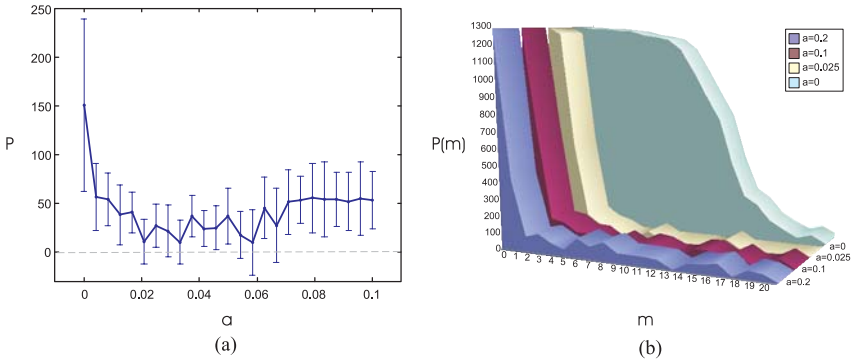

Fig. 1. Median and standard deviation of the index P vs. the parameter a (a) and an illustration of the learning process for various values of a, i.e., the performance index $P(m)$ in the consecutive learning epochs m (b).

If we consider these results and take into account that the refractory period of the presynaptic neurons is $t_{\text{ref}} = 3 \times 10^{-3}$ s, then we conclude that the best learning performance was observed in a situation where each single spike in $S^d(t)$ or $S^o(t)$ was correlated with at most one or two spikes at the particular presynaptic inputs.

Next, we search for the optimal ratio between τ_+ and τ_-. For this reason we keep τ_+ fixed and modify only τ_- in the range $[0.1\tau_+, 8\tau_+]$. We perform three series of simulations for $A_- = 0.1A_+$, $A_- = 0.5A_+$ and $A_- = A_+$, respectively (Fig. 4). In all the three cases we observe that the learning performance deteriorates as the contribution of the negative part to the learning window increases.

This is particularly evident for $A_- = 0.5A_+$ and $A_- = A_+$, where the relationship between P and τ_-/τ_+ is almost linear. However, even for the case of $A_- = 0.1A_+$, where the contribution of the negative part of the learning window is already insignificant due to the small value of A_-, we still observe the same tendency, i.e., the learning performance decreasing with an increase in τ_-/τ_+.

These results confirm our previous observation that, in order to improve the learning performance, it is reasonable to reduce the contribution of the negative part of the learning window in the ReSuMe learning algorithm.

3.4. Shape of the learning window. So far our analysis has been restricted to experiments with the exponential learning window. This choice was motivated mainly by experimental findings indicating that the exponential function can serve as a good mathematical model of the synaptic plasticity and of the relationship between the pre-/postsynaptic firing delays and the synaptic efficacy changes (Markram, Luebke, Frotscher and Sakmann, 1997; Bi, 2002).

On the other hand, it is intriguing to determine to what extent the shape of the learning window could influence the convergence of the ReSuMe learning process.

To answer this question here we examine six learning windows: a rectangular window, two versions of the linear windows, the original (exponential) window and two variants of double-exponential windows.

The rectangular and linear windows are examined mainly to find out whether the mathematical description of the learning window can be simplified without affecting the convergence of learning. This is an important issue in the context of possible hardware implementations of the ReSuMe learning algorithm (Kasiński and Kraft, 2006; Kraft, Kasiński and Ponulak, 2006).

The double-exponential shape is sometimes suggested as another approximation of learning windows observed in biological synapses. A thorough analysis of such a shape of the learning window was performed in the context of STDP processes, e.g., in (Kempter, Gerstner and van Hemmen, 1999; van Hemmen, 2001). Here, we investigate in particular the applicability of the double-exponential window to the ReSuMe learning method.

We performed a series of experiments in which we examined the learning process by implementing the particular learning windows in the ReSuMe learning rule.

For all learning windows considered the experimental procedure was identical, i.e., we trained a single neuron having 1600 inputs. Each synaptic input was assumed to deliver a single input spike at a randomly chosen time. For each learning window we used the same target spike pattern to be reconstructed. The training was performed for 40 epochs.

In order to be able to compare the results observed for the particular learning windows, the parameters were set in a way to ensure that the integrals of the particular windows over time were in all cases similar. This approach was motivated by the assumption that the statistical influence (i.e., for the very large number of spikes) of the two different learning windows on the given synaptic weight should be similar if the areas under the particular learning windows are the same. Thus, if the compared learning windows can ensure the learning convergence, then we expect the learning process to have a similar evolution. The

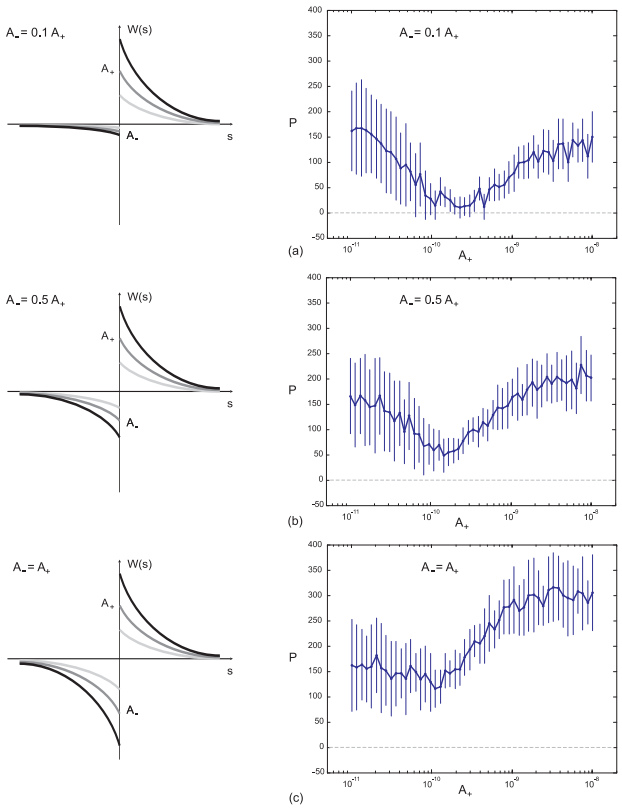

Fig. 2. Median and standard deviation of the P index vs. the amplitude of the learning window. Three cases are considered, with the amplitude of the negative part of the learning window (a) $A_- = 0.1A_+$, (b) $A_- = 0.5A_+$ and (c) $A_- = A_+$. The left-hand panels illustrate the shapes of the learning windows for the particular cases.

details of the experimental procedure are presented in Appendix.

In the following we discuss and compare the results obtained for the particular learning windows.

3.4.1. Rectangular learning window. In the case of the rectangular learning window it is assumed that the amplitude of the weight change is constant as long as the time delay s between the presynaptic and postsynaptic firings (or the presynaptic and target firings) falls within some given time range. The rectangular learning window is described by a simple mathematical formula:

$$W(s) = \begin{cases} +A_+ & \text{if } s \in [0, \tau_+], \\ -A_- & \text{if } s \in [\tau_-, 0), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

The learning window is illustrated in Fig. 7(a) (left-hand side panel). In the diagram of the performance index $P(m)$ (left-hand side panel of Fig. 7(c)) one can observe a significant learning improvement at the initial stage of

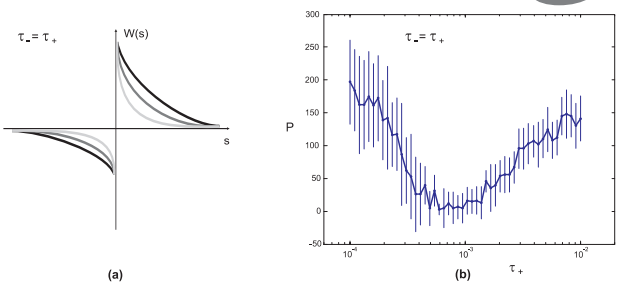

Fig. 3. Median and standard deviation of the index P as a function of time constants of the learning window, with $\tau_- = \tau_+$ (b). The left panel (a) illustrates the shapes of the learning windows for various values of τ_+.

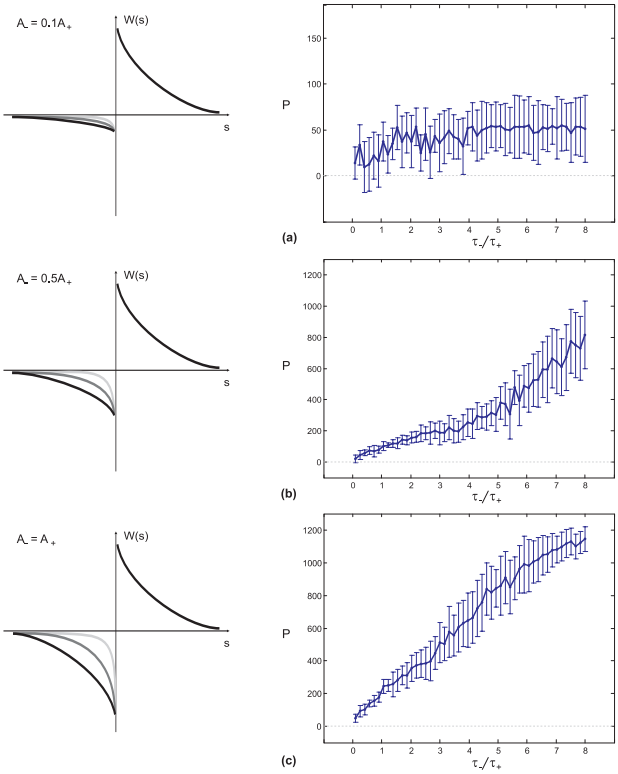

Fig. 4. Median and standard deviation of the index P plotted vs. particular ratios of the time constants τ_-/τ_+. Simulations performed with the amplitudes (a) $A_- = 0.1A_+$, (b) $A_- = 0.5A_+$, (c) $A_- = A_+$. The left-hand side panels illustrate the shapes of the learning windows for various values of τ_-/τ_+.

training (during the first four epochs). However, in the following epochs, $P(m)$ settles around a certain positive value and does not converge to zero even if the learning continues. Indeed, if we compare the resulting spike train $\hat{S}^o(t)$ at the postsynaptic neuron to the target pattern $S^d(t)$ (Fig. 7(b), left-hand side panel), we see that all spikes of $S^d(t)$ are reproduced at the neurons output, yet the precision of approximation is limited to the width of the learning window (i.e., to the range $[-\tau_-, \tau_+]$). In order to ex-

plain this phenomenon, consider a scenario with the single input, output and target firings at $t^{\rm in}$, $t^{\rm out}$ and t^d, respectively (Fig. 5). In this case the synaptic weight is modified if and only if exactly one of the postsynaptic or target spikes falls within a time range $[(t^{\rm in}-\tau_-),\ (t^{\rm in}+\tau_+)]$ of the learning window (see Figs. 5(a) and (b)). The learning stops as soon as both spikes at $t^{\rm out}$ and $t^{\rm d}$ occur in $[(t^{\rm in}-\tau_-),\ 0]$ or $[0,\ (t^{\rm in}+\tau_+)]$, since in such cases the amplitudes of the positive and negative weight changes are the same and they balance each other (cf. Fig. 5(c)). Alternatively, there is no learning at all if both spikes at $t^{\rm out}$ and $t^{\rm d}$ are outside the learning window. This explains why the learning process does not fully converge and the precision of learning is bounded to the width of the learning window.

3.4.2. Linear learning window, Version 1. We consider two options of the linear learning window. First, we concentrate on the window defined by the following formula:

$$W(s)=\begin{cases} A_+\cdot\dfrac{s}{\tau_+} & \text{if } s\in[0,\tau_+],\\ A_-\cdot\dfrac{s}{\tau_-} & \text{if } s\in[-\tau_-,0),\\ 0 & \text{otherwise.}\end{cases} \tag{5}$$

The shape of the learning window and the results of training with the use of this window are illustrated in Figs. 7(a)–(c) (central panels). As could be expected, this shape of the learning window does not lead to the convergence of learning. Although the performance index initially decreases, it stays at a reasonably high level as the learning process continues, indicating poor approximation of the target pattern (Fig. 7(c), centre). Indeed, the resulting spike sequence at the output $\hat{S}^o(t)$ differs evidently from $S^d(t)$ (Fig. 7(b), centre).

The effect of the analysed learning window on the weight changes is opposite to what is expected in order to move the output spike towards $t^o\approx t^d$ (for illustration see Figs. 6(a) and (b)). This is the reason why this learning window does not guarantee the convergence of the learning process.

3.4.3. Linear learning window, Version 2. Here, we consider another version of the linear window. This time the maximal or minimal values of $W(s)$ are reached for s close to zero and the slope of the window's linear parts is negative (Fig. 7(a), right). The learning window is described by the following formula:

$$W(s)=\begin{cases} A_+\cdot\left(1-\frac{s}{\tau_+}\right) & \text{if } s\in[0,\tau_+],\\ A_-\cdot\left(1-\frac{s}{\tau_-}\right) & \text{if } s\in[-\tau_-,0),\\ 0 & \text{otherwise.}\end{cases} \tag{6}$$

In contrast to the previous example, the learning window described by (6) implies weight changes that lead to the convergence of the learning process (Figs.6(c) and (d)). The analysis of the typical learning result with the window given by (6) confirms good learning performance and the stability of the optimal solution (Figs. 7(b) and (c), right). However, unlike in the case of the exponential learning window, the convergence cannot be formally proved (for details, we refer to (Ponulak, 2006a)).

3.4.4. Exponential learning window. We rewrite here the formula describing the exponential learning window:

$$W(s)=\begin{cases} +A_+\cdot\exp\left(\frac{-s}{\tau_+}\right) & \text{if} \quad s\geq 0,\\ -A_-\cdot\exp\left(\frac{s}{\tau_-}\right) & \text{if} \quad s<0.\end{cases} \tag{7}$$

The shape of this window is illustrated in Fig. 7(d) (left panel). The quality of the learning process and the learning results are comparable to the ones observed for the linear learning window given by (6) (compare Figs.7(e) and (f) (left) and Figs.7(b) and (c) (right)).

Note, that the influence of the linear window in (6) on the synaptic weights is sharply restricted to the time range $s\in(-\tau_-,\tau_+)$ and for any s outside this range there is no learning effect. By contrast, the exponential window in (7) covers, in principle, the whole time domain and may affect the synaptic weights, gradually leading to the convergence even for long delays between the presynaptic- and postsynaptic- or target spikes.

3.4.5. Double-exponential learning window. Finally, we consider the double-exponential learning windows

$$W(s)=\begin{cases} +A_+\cdot\left[\exp\left(\frac{-s}{\tau_{r+}}\right)-\exp\left(\frac{-s}{\tau_{d+}}\right)\right] & \text{if } s\geq 0,\\ -A_-\cdot\left[\exp\left(\frac{s}{\tau_{r-}}\right)-\exp\left(\frac{s}{\tau_{d-}}\right)\right] & \text{if } s<0,\end{cases} \tag{8}$$

with τ_{r+},τ_{r-} and τ_{d+},τ_{d-} being the time constants of the rising and decaying slopes of $W(s)$, respectively. As usually, the subscript '$+$' ('$-$') is used to describe the parameters of the positive (negative) part of the window.

In our experiments we consider two cases: the one with a very short rising slope of the double-exponential window (Fig. 7(d), centre) and yet another one with a relatively longer rising part of the window (Fig. 7(d), right).

In the first case the following time constants are assumed: $\tau_{r+}=\tau_{r-}=0.14\times10^{-3}$ s and $\tau_{d+}=\tau_{d-}=2\times10^{-3}$ s. From the learning performance diagram we see that during learning the index $P(m)$ decreases quickly to zero with only slight fluctuations (Fig. 7(f), centre). The target spiking pattern $S^d(t)$ is correctly reconstructed at the output neuron ($\hat{S}^o$) with a high precision (Fig. 7(e), middle).

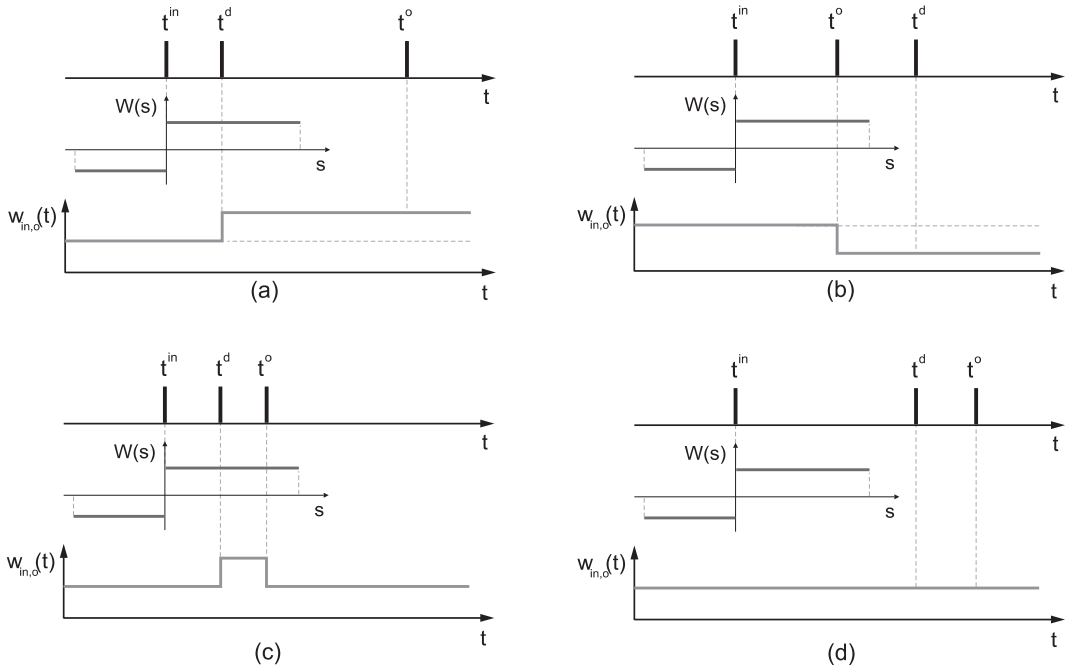

Fig. 5. Learning with the rectangular learning window. The synaptic weights are modified if either t^d or t^o falls within the learning window (a), (b). In other cases, the total weight change equals zero (c), (d).

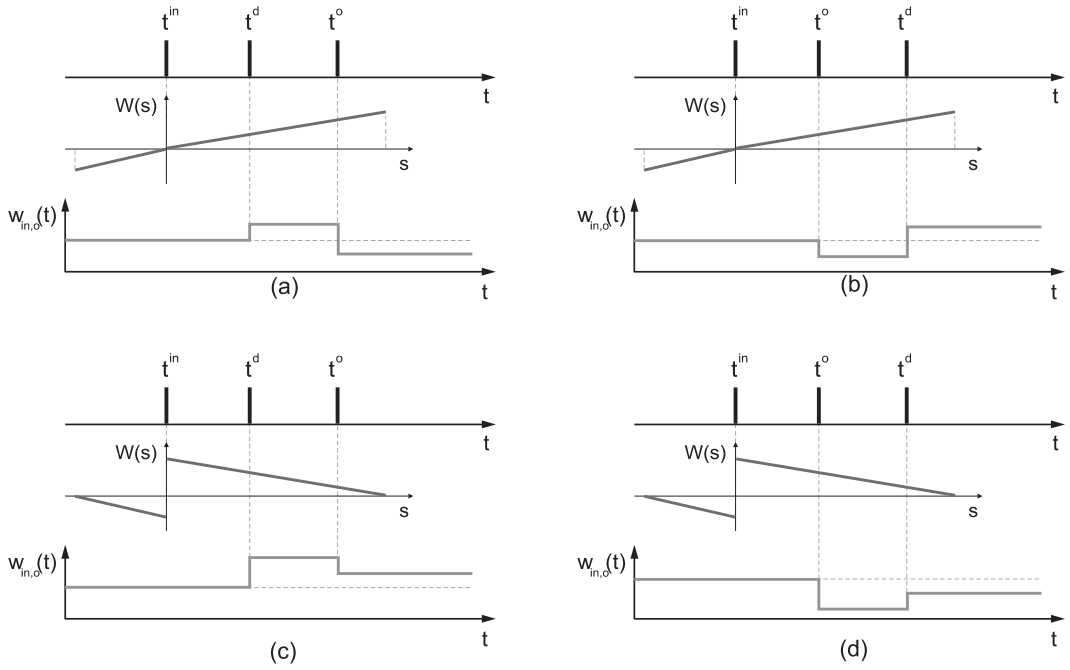

Fig. 6. Learning with the linear learning windows defined by (5) ((a) and (b)) and by (6) ((c) and (d)). Whenever the target spike at t^d precedes the postsynaptic spike at t^o (as illustrated by the upper panels), a positive total weight change is required to force the postsynaptic neuron to fire earlier. Contrary (bottom panels), in order to force the neuron to fire later, the weight change should be negative. By comparing (a)–(d) we observe that the expected weight changes are performed only by the learning window defined by (6) ((b) and (d)).

The time constants of the second learning window are $\tau_{r+} = \tau_{r-} = 0.3 \times 10^{-3}$ s and $\tau_{d+} = \tau_{d-} = 2 \times 10^{-3}$ s. In this case we observe that the approximation quality is improved during the first few learning epochs. However, as the training continues, the performance index $P(m)$ increases, indicating a decline in the approximation quality (Fig. 7(f), right). We observe that the learning process does not finally converge and the resulting output signal $\hat{S}^o$ differs from the target signal $S^d(t)$ (Fig. 7(e), right).

The difference between the learning results observed in both cases can be explained on the basis of our theoretical deliberations presented in (Ponulak, 2006a). According to that analysis we note that the learning convergence relies only on the decaying part of the learning window. However, if the contribution of the rising part of the learning window is sufficiently small, then its diverging effect on the learning process is suppressed by the desired action of the decaying part. This is the case, e.g., for the first double-exponential learning window presented in Fig. 7(d) (middle). A thorough analysis of this case revealed that only about 5% of all 1600 presynaptic spikes fall in the rising part of the learning window, to correlate with the spikes in $S^o(t)$ or $S^d(t)$. However, in the case of the double-exponential window illustrated in Fig. 7(d) (right), where the rising slope amounts to about 15% of the total width of the learning window, the contribution of the rising part in the learning processes increased to 25%. This affected the learning results substantially, leading to the lack of convergence.

In this context we see that the double-exponential learning windows, which are sometimes considered as a good approximation of the biologically realistic learning windows observed for STDP, can be successfully applied to the ReSuMe training only if some conditions on the window time constants are satisfied.

4. Conclusions

In this paper we presented results of a parametric analysis of the ReSuMe learning process. The goal of this study was to determine the influence of individual parameters on the learning convergence. Our analysis leads to the following conclusions:

- The role of the non-Hebbian contribution (parameter a in Eqn. (1) to the ReSuMe learning is to set the activity (the instantaneous firing rate) of the postsynaptic neuron to the level determined by the desired signal, but without taking into account a precise timing of the particular spikes (cf. Section 3.1).

- The influence of the negative part of the learning window $W(s)$ (cf. Eqn. (1)) on the learning process is disadvantageous. A reduction or even a rejection of this part of the learning window leads to a significant improvement of the learning performance in ReSuMe (cf. Sections 3.2 and 3.3).

- The best learning performance was observed when the time constants of the learning window were set such that each single spike in $S^d(t)$ or $S^o(t)$ was correlated with at most one or two spikes at the particular presynaptic inputs. This is the case if the effective range of the learning window is of the order of the absolute refractory period of the trained neuron (cf. Section 3.3).

- The learning convergence is observed only for the learning windows which are described or can be approximated well by monotonically decreasing functions of a time distance between the presynaptic and the postsynaptic or target spikes (cf. Section 3.4). This observation confirms our previous theoretical results discussed in (Ponulak, 2006a), where we applied a formal mathematical approach to investigate the convergence of learning with ReSuMe.

Acknowledgements

The work was partially supported by the Polish Ministry of Science and Higher Education, project no. 1445/T11/2004/27.

References

Bi G.-Q. (2002). Spatiotemporal specificity of synaptic plasticity: Cellular rules and mechanisms, *Biological Cybernetics* **87**: 319–332.

CSIM (2002). CSIM: *A neural circuit SIMulator*. The IGI LSM Group, Technical University, Graz, `http://www.lsm.tugraz.at`.

Freeman J. A. and Skapura D. M. (1991). *Neural Networks Algorithms, Applications, and Programming Techniques*, Addison-Wesley, Redwood City, CA.

Gerstner W. and Kistler W. (2002a). Mathematical formulations of Hebbian learning, *Biological Cybernetics* **87**(5–6): 404–415.

Gerstner W. and Kistler W. (2002b). *Spiking Neuron Models. Single Neurons, Populations, Plasticity*, Cambridge University Press, Cambridge.

Hertz J., Krogh A. and Palmer R. (1991). *Introduction to the Theory of Neural Networks*, Addison-Wesley, Redwood City, CA.

Kangas J. and Kohonen T. (1996). Developments and applications of the self-organizing map and related algorithms, *Mathematics and Computers in Simulation* **41**(1): 3–12(10).

Kasiński A. and Kraft M. (2006). The design of a compact LIF-neuron circuit in FPGA to enable implementation of large-scale spiking neuron networks with learning capabilities,

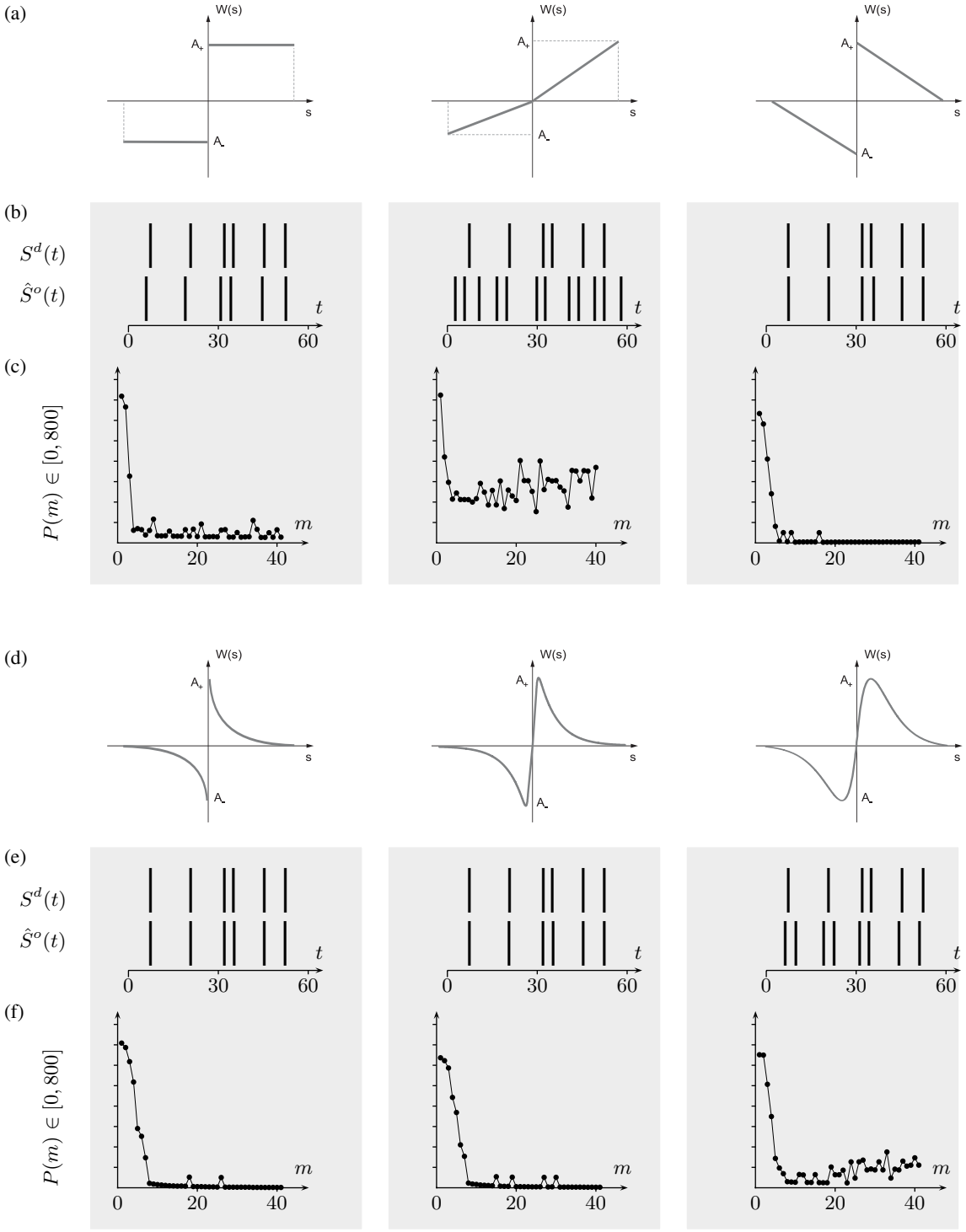

Fig. 7. Illustration of the different shapes of the learning windows ((a) and (d)) and their influence on the performance of the learning process ((b), (c), (e), (f)). Rectangular window ((a), (b), (c), left). Linear window defined by (5) ((a), (b), (c), centre). Linear window defined by (6) ((a), (b), (c), right). Exponential window ((d), (e), (f), left). Double-exponential windows ((d), (e), (f), centre and right).

Proceesings of the International Conference on Artificial Intelligence and Soft Computing, ICAISC'2006, Warsaw, Poland, pp. 57–64.

Kasiński A. and Ponulak F. (2005). Experimental demonstration of learning properties of a new supervised learning method for the spiking neural networks, *Lecture Notes in Computer Science*, Vol. 3696, pp. 145–153.

Kempter R., Gerstner W. and van Hemmen J. L. (1999). Hebbian learning and spiking neurons, *Physical Review E* **59**(4): 4498–4514.

Korbicz J., Obuchowicz A. and Uciński D. (1994). *Artificial Neural Networks: Foundations and Applications*, Akademicka Oficyna Wydawnicza PLJ, Warsaw. (in Polish).

Kraft M., Kasiński A. and Ponulak F. (2006). Design of the spiking neuron having learning capabilities based on FPGA circuits, *Proceedings of the 3rd International IFAC Workshop on Discrete-Event System Design*, Rydzyna, Poland, pp. 301–306.

Maass W. and Bishop C. (Eds.) (1999). *Pulsed Neural Networks*, The MIT Press, Cambridge, M.A.

Maass W., Natschlaeger 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.

Markram H., Luebke J., Frotscher M. and Sakmann B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs, *Science* **275**(5297): 213–215.

Natschlaeger T., Maass W. and Markram H. (2002). The "liquid computer", a novel strategy for real-time computing on time series, *Foundations of Information Processing of TELEMATIK* **8**(1): 32–36.

Papik K., Molnar B., Schaefer R., Dombovari Z., Tulassay Z. and Feher J. (1998). Application of neural networks in medicine—A review, *Medical Science Monitor* **4**(3): 538–546.

Ponulak F. (2005). ReSuMe—New supervised learning method for Spiking Neural Networks, *Technical Report*, Institute of Control and Information Engineering, Poznań University of Technology. Available at `http://d1.cie.put.poznan.pl/~fp/`.

Ponulak F. (2006a). ReSuMe—Proof of convergence, *Technical Report*, Institute of Control and Information Engineering, Poznan University of Technology. Available at `http://d1.cie.put.poznan.pl/~fp/`.

Ponulak F. (2006b). *Supervised Learning in Spiking Neural Networks with ReSuMe Method*, Ph.D. thesis, Institute of Control and Information Engineering, Poznań University of Technology. Available at: `http://d1.cie.put.poznan.pl/~fp/`.

Ponulak F., Belter D. and Kasiński A. (2006). Adaptive central pattern generator based on spiking neural networks, *Proceedings of EPFL LATSIS Symposium 2006, Dynamical Principles for Neuroscience and Intelligent Biomimetic Devices*, Lausanne, Switzerland, pp. 121–122.

Ponulak F. and Kasiński A. (2005). A novel approach towards movement control with spiking neural networks, *Proceedings of the 3rd International Symposium on Adaptive Motion in Animals and Machines*, Ilmenau, Germany. (Abstract).

Ponulak F. and Kasiński A. (2006a). Generalization Properties of SNN Trained with ReSuMe, *Proceedings of the European Symposium on Artificial Neural Networks, ESANN'2006*, Bruges, Belgium, pp. 623–629.

Ponulak F. and Kasiński A. (2006b). ReSuMe learning method for spiking neural networks dedicated to neuroprostheses control, *Proceedings of EPFL LATSIS Symposium 2006, Dynamical Principles for Neuroscience and Intelligent Biomimetic Devices*, Lausanne, Switzerland, pp. 119–120.

van Hemmen J. (2001). Theory of synaptic plasticity, *in* F.Moss and S.Gielen (Eds.), *Handbook of Biological Physics, Neuro-informatics, Neural Modelling*, Elsevier, Amsterdam, Vol. 4, pp. 771–823.

Appendix

Spike train. Let t_m^f denote the firing time of neuron m (where $f = 1, 2, \ldots$ is a label of each individual spike emitted by this neuron). Similarly to (Gerstner and Kistler, 2002b), we define a spike train of a neuron m as a sequence of the impulses triggered at the firing times:

$$S_m(t) = \sum_f \delta(t - t_m^f),$$

where $\delta(x)$ is the Dirac function with $\delta(x) = 0$ for $x \neq 0$ and $\int_{-\infty}^{\infty} \delta(x)\,\mathrm{d}x = 1$.

Neuron model: All neurons considered in the described simulations were modelled as Leaky-Integrate-and-Fire (LIF) units (Gerstner and Kistler, 2002b). The membrane potential u_m of a neuron is given by

$$\tau_m \frac{\mathrm{d}u_m}{\mathrm{d}t} = -(u_m - E_m) + R_m \cdot \Big(\sum i_{\mathrm{syn}}(t) + i_{\mathrm{ns}} \Big), \quad (9)$$

where $\tau_m = C_m \cdot R_m$ is the membrane time constant, C_m and R_m are the membrane conductance and resistance, respectively, E_m is a membrane potential at rest, $\sum i_{\mathrm{syn}}(t)$ is the sum of the currents supplied by the particular synapses entering a given neuron, i_{ns} is the sum of a non-specific background current and a Gaassion random variable with zero mean and given variance noise. At time $t = 0$ the membrane potential is set to u_{init}. If u_m exceeds the threshold voltage ϑ, it is reset to u_{res} and held there for the length t_{ref} of the absolute refractory period.

Model of the synaptic response: We implement a synaptic response as $i_{\mathrm{syn}}(t) = w \cdot \exp(-\tau_d/t)$ for each spike which hits the synapse at time t with an amplitude of w and a

decay time constant of τ_d. It is assumed that the responses of all the spikes are added up linearly.

Performance index: In order to quantitatively estimate the error between the desired and the output spike trains ($S^d(t)$ and $S^o(t)$, respectively), we define the performance index $P(m)$ as

$$P(m) = \int \left| L\big(S^d(t)\big) - L(S^o(t)) \right| \mathrm{d}t, \qquad (10)$$

where, for any spike train $S(t)$, $L(S(t))$ denotes a lowpass filtering:

$$L(S(t)) = \sum_f \exp\left(\frac{-t + t^f}{\tau}\right) \cdot H\big(t - t^f\big). \qquad (11)$$

Here $H(x) = 0$ for $x < 0$, and $H(x) = 1$ otherwise; τ is a filter time constant.

According to (10), the performance index takes on high values if $S^o(t)$ and $S^d(t)$ differ significantly, while it decreases to zero for $S^o(t) = S^d(t)$.

NMC neurons parameters: $C = 0.2$ μF, $R = 1$ MΩ, $t_{\mathrm{ref}} = 0.003$ s, $\vartheta = -0.057$ V, $u_{\mathrm{init}} = -0.058$ $\pm 1\%$ V, $E_m = -0.060$ V, $u_{\mathrm{res}} = -0.062$ $\pm 1\%$ V, $i_{\mathrm{ns}} = [0,\ 5 \cdot 10^{-13}]$ A.

Readout neurons parameters $C = 1$ nF, $R = 1$ MΩ, $t_{\mathrm{ref}} = 0.0025$ s, $\vartheta = -0.055$ V, $u_{\mathrm{init}} = -0.058$ $\pm 1\%$ V, E_m=-0.060 V, $u_{\mathrm{res}} = -0.062$ $\pm 1\%$ V, $i_{\mathrm{ns}} = [0,\ 5 \times 10^{-13}]$ A.

Network structure: number of inputs $= 1$, total number of neurons in NMC $= 1600$, number of microcolumns in NMC $= 2$, number of outputs (learning neurons) $= 1$.

Connections from the inputs to the NMC: fraction of the excitatory connections $f_{\mathrm{exc}} = 100\%$, strength of the synaptic connections $w = [0,\ 2] \times 10^{-8}$, probability of a connection[2]: $\lambda = +\infty$ and $C_{\mathrm{scale}} = 10$, synaptic delay $d = [0,\ 0.004]$ s, synaptic time constant $\tau_d = 0.003$ s.

Connections within the particular microcolumns: size of a column: $8 \times 5 \times 20$, f_{exc} =75%, $w = [-2,\ 2] \times 10^{-8}$, probability of a connection: $\lambda = 2.4$ and $C_{\mathrm{scale}} = 10$, $d = [0,\ 0.004]$ s, $\tau_d = 0.003$ s.

Connections from the NMC to the readouts: $f_{\mathrm{exc}} = 80\%$, $w = [-2,\ 2] \times 10^{-8}$, probability of a connection: $\lambda = +\infty$ and $C_{\mathrm{scale}} = +\infty$, $d = 0$ s, $\tau_d = 0.001$ s.

Learning rules (default parameters): non-Hebbian rate of the weight change $a = 0.01$, amplitude of the positive part of the learning windows $A_+ = 4 \times 10^{-10}$, amplitude of the negative part of the learning windows $A_- = 1 \times 10^{-10}$, time constant of the positive part of the learning windows $\tau_+ = 0.002$ s, time constant of the negative part of the learning windows $\tau_- = 0.002$ s.

Rectangular learning window (cf. Eqn.(4)): $a = 0.001$, $A_+ = 10 \times 10^{-11}$, $A_- = 1 \times 10^{-11}$, $\tau_+ = \tau_- = 0.006$ s.

Linear learning windows (cf. Eqn. 5 and 6): $a = 0.001$, $A_+ = 20 \times 10^{-11}$, $A_- = 2 \times 10^{-11}$, $\tau_+ = \tau_- = 0.006$ s.

Exponential learning window (cf. Eqn. 7): $a = 0.001$, $A_+ = 40 \times 10^{-11}$, $A_- = 4 \times 10^{-11}$, $\tau_+ = \tau_- = 0.002$ s.

Double-exponential learning windows (cf. Eqn. 8): $a = 0.001$, $A_+ = 40 \times 10^{-11}$, $A_- = 4 \times 10^{-11}$, rise time constant $\tau_{r+} = \{14, 30\} \times 10^{-5}$ s, rise time constant $\tau_{r-} = \{14, 30\} \times 10^{-5}$ s, decay time constant $\tau_{d+} = 0.002$ s, decay time constant $\tau_{d-} = 0.002$ s.

Received: 10 July 2007
Revised: 22 November 2007

[2]The probability of a synaptic connection from neuron a to neuron b, and from b to a, is defined as $c \cdot \exp(-D(a,b)^2/\lambda)$, where c and λ are positive constants and $D(a,b)$ is the Euclidean distance between the neurons a and b. Depending on whether the neurons a and b are excitatory (E) or inhibitory (I), the value of c is 0.3 (EE), 0.2 (EI), 0.4 (IE), 0.1 (II). The parameter C_{scale} specifies how to scale the overall connection probability (see (Maass *et al.*, 2002; CSIM, 2002) for details).