1 Summary

In [1], the authors propose a novel approach based on the perceptron rule to learn precisely timed spikes. The main goal is to design a neural network that is capable of learning, or to put it in another way, reproducing, a given set of spike trains. More specifically, suppose we have an N-to-1 network, i.e., N pre-synaptic neurons and 1 post-synaptic one. Then, we would like to find the connectivity vector \( w \) in such a way that the network could map a given set of input and output spike trains.

The problem is tackled from theoretical point of view, i.e., what are the requirements on the spike trains and how many such associations can be learned, as well as practical aspects, i.e., what happens when more realistic neural models are used. As a side application, the proposed idea is used to infer the synaptic connectivity of a set of neural networks.

2 Method’s Summary

2.1 Feed forward networks

The authors first consider a single-layer feed forward network of Leaky Integrate and Fire (LIF) neurons. The network constitutes \( N \) presynaptic (afferent) neurons. Each afferent \( i \) emits spikes at certain times \( t_i \), which affect the membrane potential of the post-synaptic neuron.

<table>
<thead>
<tr>
<th>Point to consider</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note that the links do not have random delays.</td>
</tr>
</tbody>
</table>

The authors use a brilliant trick to transform the set of non-linear update equations (in \( w \)) to a system of linear equations. The proposed trick is very similar to the one considered in [2]. The authors have also developed a technique, called High-Threshold Projection (HTP) that enables neurons to solve these equations.

<table>
<thead>
<tr>
<th>Idea for future</th>
</tr>
</thead>
<tbody>
<tr>
<td>The authors use a brilliant trick to transform the set of non-linear update equations (in ( w )) to a system of linear equations.</td>
</tr>
</tbody>
</table>

The proposed technique is based on the Perceptron rule and essentially, makes sure that the equality constraint \( U(t_d) = U_{\text{thr}}, \forall t_d \) and the inequality constraint \( U_t < U_{\text{thr}}, \forall t \neq t_d \) are satisfied.
In other words, it makes sure that the post-synaptic neuron fires at exactly the times it should fire \((t_d)\) and remains silent at all other times.

The authors have also proposed another approach for dealing with finite-precision spike trains, which will more appropriate in practical situations.

**Idea for future**

The finite precision approach could also be a good technique to measure the accuracy of our proposed approach to infer the neural connectivity graph.

### 2.2 Recurrent networks

The authors then turn to recurrent networks where the goal is to train a set of recurrently-connected neurons (with some afferents to perform the stimulation role) in such a way that they learn to produce a specific spike train (with finite-precision of course) in response to a particular input spiking pattern.

The authors show that using the Finite Precision learning algorithm, where the weights are updated according to the perceptron rule but only for the first occurring spiking mismatch, the recurrent network is able to reproduce a *periodic* spike pattern.

**Point to consider**

Note that the links do not have random delays.

### 3 Results

**Strong result**

The authors prove that the proposed Perceptron-based approach for a single layer feed forward network "converges after a finite number of updates to a set of weights that implements exactly the desired input-output spike time mapping if such a solution exists" [1].

**Strong result**

The authors prove that if there exists a solution, the finite-precision approach for a single layer feed forward network also convergence to the solution in finite time. Furthermore, it is robust to small perturbations in the spike times.

**Strong result**

The authors find out that for large single layers feed forward networks the capacity is proportional to the number of input synapses.
The authors show that using the Finite Precision learning algorithm, where the weights are updated according to the perceptron rule but only for the first occurring spiking mismatch, the recurrent network is able to reproduce a periodic spike pattern.

**Question for future**

I wonder if the results could be extended to non-periodic output spiking patterns as well, which would be fantastic!

### 4 Method’s Details and Ideas

The authors start with a linear model for neurons and with a brilliant trick account for the effect of membrane potential reset after firing an action potential. However, for this trick to work, the following assumptions should hold for the post-synaptic spike times $\{t_d\}$:

1. $U(t_d) = U_{thr}$, for all $t_d$
2. $\frac{dU}{dt}|_{t_d} > 0$, for all $t_d$
3. $U(t) < U_{thr}$ at all times except $t_d$

In the above, $U(t)$ is the membrane potential of the post-synaptic neuron at time $t$ and $U_{thr}$ is the firing threshold.

**Question for future**

I wonder how realistic the above assumptions are in reality, especially the third one. On the face of it, this seems to be reasonable valid in general to me. If so, this additional information could become very useful.

Using the above assumptions, the problem of learning the vector $w$ becomes equivalent to solving a set of linear equations. More specifically, we replace $t$ by $t_d$ and consider

$$U(t_d) = U_{thr} = w^\top x(t_d) - U_{thr} x_{\text{reset}}(t_d),$$

where $x(t)$ is the vector of pre-synaptic spike trains and

$$x_{\text{reset}}(t) = \sum_{t_{\text{spike}}<t} u_r(t - t_{\text{spike}}) = \sum_{t_{\text{spike}}<t} e^{-(t-t_{\text{spike}})/\tau_m}$$

Having the above equations, and assuming that we have $n_{\text{spikes}}$ output spikes (at times $\{t_d\}$), the above equations result in a set of linear equations. Assuming that the vectors $x(t_d)$ are linearly independent, the solution of this set of equations is given by

$$w = (I - X(X^\top X)^{-1}X^\top)(1 + x_{\text{reset}})U_{thr}$$

Note that while the proposed method (HTP) is essential for the main objective of this paper, namely, enabling a set of neurons learn spikes association, for other purposes, such as inferring the connectivity matrix of the graph it might not be necessary, as one could use more sophisticated techniques to solve the set of linear equations.
4.1 Capacity of LIF Neurons

The authors have investigated the capability of LIF neurons to learn the association between input and output spike trains. The capacity here refers to “the maximal combined duration, $T$, of the sequences that can be learned”.

4.2 Practical Consideration

While learning accurate spike trains is important from theoretical perspective, in practice spike timings are not necessarily accurate. This fact seriously affects the learning algorithm as we can not enforce the reset potential at specified times.

To resolve this issue, we should specify a desired precision and adapt the algorithms to work with finite precision. Then, the basic idea is to adjust the weights if the output neuron misfires or fail to fire as expected in a finite time window.

However, the finite precision means that the linearization trick anymore. As a result, the convergence proof for the previous case does not apply anymore. However, the authors have proved that the finite precision approach works as well.

4.2.1 Points to Consider for Simulations

The authors have used

- A Poisson random variable with rate $r_{in} = 10Hz$ for pre-synaptic neurons.
- $N = 1000$ pre-synaptic neurons were considered.
- The duration of each input pattern was set to be 1 second.
- The neural weights are chosen according to a log-normal distribution with parameter $\sigma = 0.97$.
- $80\%$ of all synapses are set to be excitatory while $20\%$ are inhibitory. To keep the network balanced, the weight of inhibitory neurons are chosen from a distribution five times stronger than that of excitatory neurons. These parameters are adjusted to ensure a $10Hz$ output spiking rate.
- The time coefficients of the network were set to $\tau_m = 15ms$ and $\tau_s = 5ms$.
- The synaptic input current was modeled as

$$I_{\text{synaptic}}(t) = (E_i - U(t)) \sum_{i=1}^{N} g_i \sum_{t_i < t} e^{\frac{t-t_i}{\tau_s}},$$  \hspace{1cm} (3)

where $U(t)$ is the membrane potential and $E_i$ is the reversal potential of the synapse of pre-synaptic neuron $i$.
- $E_i$ was set to 0 for excitatory connections and $-80mV$ for inhibitory ones.
- We have $g_i = w_i/\Delta$, with $\Delta = 55mV$ and $\Delta = -25mV$ for excitatory and inhibitory connection.
• To measure the spike reproduction accuracy in *Finite Precision* (FP) mode, authors require the *student* neuron to spike once within each spiking window \([t_d - \epsilon/2, t_d + \epsilon/2]\), where \(t_d\) are the spiking times for the pre-synaptic neurons.

• **The tolerance window was set to** \(\epsilon = 3\text{ms}\).

• The learning for the FP scenario is very similar to our own setting, namely, there are several *learning iterations* in each of which a pre-synaptic spike pattern is presented to the network. However, since each input pattern has a duration of 1 second, it is not exactly similar to our scenario.

• In the FP scenario, the authors update the weights in the *student* network only when the first error happens (a missed/undesired spike occur within the firing time window). In case of a missed spike, the time of error is considered to be the *end of the tolerance window*.

• 3000 input patterns were considered to learn the weights of the *feed forward network*.

• Authors used the following measure to calculate the quality of weight reconstruction:

\[
R^2 = 1 - \langle (g_{i}^{\text{student}} - g_{i}^{\text{teacher}})^2 \rangle / \text{var}(g_{i}^{\text{student}})
\]

For the recurrent network simulations, here are the main points of the algorithm

• \(N = 100\) recurrent neurons and \(N_{ext} = 10\) external driving neurons were considered.

• The duration of the periodic stimulus was set to \(T_p = 250\text{ms}\).

**Idea for future**

To measure the accuracy of the reproduced spike train, the authors use the *spike distance metric* proposed in [6], with \(q^{-1} = 0.75\text{ms}\).

**Idea for future**

To learn the weights, they were initialized *randomly*, according to the same distribution as the actual weights.

**Idea**

The network parameters are adjusted in such a way that a 10Hz output spiking rate is observed.

**Point to consider**

While the FP setting is very similar to our *stimulate-observe-rest* scenario, it differs in the fact that each input pattern has a duration (of 1s) as well. Nevertheless, it seems that after this duration network is allowed to get back to the rest state.
Idea
To learn the weights, an adaptive scenario was considered according with $A = 0.005$ and $\lambda = 0.01$ with $10^6$ iterations (pattern presentation).

Point to consider
3000 input patterns were considered to learn the weights of the feed forward network.

Idea for future
The authors have also used HH model to generate spikes. Consult the paper for the model (Section: Learning from HH teacher). A notable point is that in this scenario, they optimize $\tau_m$, $\tau_s$, and $\epsilon$ for the LIF student neuron to get the best result.

5 Related Work
This part below is directly copied from the paper and contains some very good literature review: "Our learning algorithm offers a simple method for reconstruction of synaptic weights from observed spike times. In particular, it uses a standard neuron model with a clear biophysical interpretation, incorporating full voltage reset, and synaptic currents with finite time constant and a strength characterized by a single amplitude. The simple reconstruction example shown here with feedforward networks can also be used in a recurrent circuit. Recent studies of connectivity reconstruction from spiking activity used more complex stochastic generative models for the circuit (Pillow et al., 2008; Gerwinn et al., 2010). Their biophysical interpretation is less clear, particularly since these models typically represent each synapse by a modifiable temporal filter. Other LIF-based experimental procedures for weight reconstruction are restricted to pulse-like synaptic currents (Monasson and Cocco, 2011; Van Bussel et al., 2011; Memmesheimer and Timme, 2006), assumed constant external input (Van Bussel et al., 2011), or are limited to very small networks with restrictive reset schemes (Makarov et al., 2005). The realization of FP learning by a biological system requires the presence of a supervisory signal that detects the first error produced by the network in each episode. The most plausible mechanism would be comparing the network output with an internally stored template of the desired sequence. The restriction of learning to the first error can be relaxed to multiple learning events as long as they are well separated such that nonlinear interactions between the corresponding errors are minimized. Biologically, this can be implemented by refractoriness in the error signals. Most current models of neurons computational capacity are based on averaging inputs and outputs over long time windows. These models neglect the dynamic features of neuronal integration and spiking as well as the potential coding of information in the spike times. Precisely timed patterns of spikes carrying sensory information have been experimentally found in various neural systems (Kayser et al., 2009; Jones et al., 2004; Johans- son and Birznieks, 2004; Gollisch and Meister, 2008). In the mammalian motor cortex, their occurrence correlates with internal cognitive states and task performance (Riehle et al., 1997; Putrino et al., 2010) and in the motor cortex of songbirds, they govern the song generation process (Yu and Margoliash, 1996; Leonardo and Fee, 2005). Our work shows that simple circuits of spiking neurons can robustly implement
and learn temporally precise codes under biologically realistic conditions.

References


