Ludwig-Maximilians-Universität at München
Department “Institut für TODO”
Lehr- und Forschungseinheit TODO
Prof. Dr. PROFESSORS NAME

Bachelor’s Thesis

THESIS TITLE

Valentin Herrmann
EMAIL

Bearbeitungszeitraum:	START bis END
Betreuer:	SUPERVISOR
Verantw. Hochschullehrer:	Prof. Dr. PROFESSORS NAME

Abstract

This thesis proposes ...

Contents

1 Introduction

While the hype on AI is still ongoing there are still people wondering if current AI models are fundamentally able of reasoning. Many other possible models could be better fitted to the task of reasoning. Among others Spiking neural networks present a model closer to the workings of the brain. In fact, they have been called the 3rd generation of AI models, after the 2nd generation that currently drives most successful models.

While the idea behind SNNs is quite old, they have not been as much researched, since it is much more inefficient and harder to train them. Therefore there still remain a lot of open questions about them. In this paper we shall extend on the work done in [Nguyen et al., 2025]. We will add a decaying factor to the input of the neurons and allow recursive connections between neurons in a layer.

We will roughly follow the structure of [Nguyen et al., 2025]. In the second chapter we will formally introduce discrete time leaky-integrate-and-fire SNN, d.t. LIF-SNNs. In section 3 we will give theorems about approximation of continuous functions on compact domains by d.t. LIF-SNNs. The main part will be the following section in which we will see that the number of distinct values a d.t. LIF-SNN can take on only depends on the first hidden layer and grows in particular only quadratically in time. We will further support our findings with experimental data in the last section.

2 Motivation and definition of r. d.t. LIF-SNN

2.1 Motivation

Like [Nguyen et al., 2025] we want to model the brain more closely. To this aim we will use a slightly more complex model than [Nguyen et al., 2025] did.

We have equations

$$I'(t) := -\tau_\alpha I(t) + \sum_{j=1}^{n} w_j s_j(t-\Delta t)$$
$$U'(t) := -\tau_\beta U(t) + I(t) + b - \vartheta s(t)$$

Assuming that U and I describe smoothly differentiable functions, we can now use the first-order exponential integrator method to obtain a discretization of I and U from the differential equations:

$$\begin{aligned}
&e^{-\tau_\alpha t_n}\left(I(t_{n+1}) - e^{-\tau_\alpha h} I(t_n)\right) \\
&= \int_{t_n}^{t_{n+1}} \frac{d}{dt}\left(e^{-\tau_\alpha t} I(t)\right) dt \\
&= \int_{t_n}^{t_{n+1}} e^{-\tau_\alpha t}(I'(t) - \tau_\alpha I(t)) dt \\
&= \int_{t_n}^{t_{n+1}} e^{-\tau_\alpha t} \sum_{j=1}^{n} w_j s_j(t-\Delta t) dt
\end{aligned}$$

If we assume the input from the spikes of other neurons to be constant during $[t_n, t_{n+1}]$, we get

$$\begin{aligned}
&= -\frac{1}{\tau_\alpha}(e^{-\tau_\alpha t_{n+1}} - e^{-\tau_\alpha t_n}) \sum_{j=1}^{n} w_j s_j(t-\Delta t) \\
&= \frac{1}{\tau_\alpha} e^{-\tau_\alpha t_n}(1 - e^{-\tau_\alpha h}) \sum_{j=1}^{n} w_j s_j(t-\Delta t)
\end{aligned}$$

So we get

$$I(t_{n+1}) = e^{-\tau_\alpha h} I(t_n) + \frac{1}{\tau_\alpha}(1 - e^{-\tau_\alpha h}) \sum_{j=1}^{n} w_j s_j(t-\Delta t)$$

Let us define $h := 1$ and $\alpha := e^{-\tau_\alpha}$. We get

$$I(t_{n+1}) = \alpha I(t_n) + \sum_{j=1}^{n} w_j s_j(t-\Delta t)$$

By absorbing $\frac{1-\alpha}{\tau_\alpha}$ into the weights $(w_j)_{j\in[n]}$. We obtain similarly

$$U(t_{n+1}) = \beta U(t_n) + I(t) + b - \vartheta s(t)$$

by using the first-order exponential integrator method, defining $\beta := e^{-\tau_\beta}$ and absorbing $\frac{1-\beta}{\tau_\beta}$ into $I(0)$, $(w_j)_{j\in[n]}$, b and ϑ.

2.2 Definitions

Our type of SNN should be thought of as a composition of an initial input layer, a number of hidden spiking layers with internal state and an affine-linear layer mapping spikes activations over time, so called spike trains, to the value of the output layer.

Like motivated we are adding additional structure to the definitions [Nguyen et al., 2025]. We shall first define the input $i^{[l]}(t)$, the membrane potential before spike $p^{[l]}(t)$, the membrane potential after spike $u^{[l]}(t)$ and the spike activations $s^{[l]}(t)$ of the hidden layers:

Definition 2.1. The **input vector** $i^{[l]}(t) \in \{0,1\}^{n_l}$, the **spike vector** $s^{[l]}(t) \in \{0,1\}^{n_l}$ and the **membrane potential vector** $u^{[l]}(t)$ of a hidden layer $\lambda = (W^{[l]}, b^{[l]}, u^{[l]}(0), i^{[l]}(0), \alpha^{[l]}, \beta^{[l]}, \vartheta^{[l]})$, $l \in [L]$, are recursively defined as

$$i^{[l]}(t) := \alpha^{[l]} i^{[l]}(t-1) + W^{[l]} s^{[l-1]}(t) + V^{[l]} s^{[l]}(t-1) \tag{1}$$

$$p^{[l]}(t) := \beta^{[l]} u^{[l]}(t-1) + i^{[l]}(t) + b^{[l]} \tag{2}$$

$$s^{[l]}(t) := H(p^{[l]}(t) - \vartheta 1_{n_l}) \tag{3}$$

$$u^{[l]}(t) := p^{[l]}(t) - \vartheta s^{[l]}(t) \tag{4}$$

with $s^{[l]}(0) = 0$ and given

- **initial membrane potential**: $u^{[l]}(0) \in \mathbb{R}^{n_l}$,
- **initial input**: $i^{[l]}(0) \in \mathbb{R}^{n_l}$,
- **weight matrices**: $W^{[l]} \in \mathbb{R}^{n_l \times n_{l-1}}$, $V^{[l]} \in \mathbb{R}^{n_l \times n_l}$,
- **bias vectors**: $b^{[l]} \in \mathbb{R}^{n_l}$,
- **leaky terms**: $\alpha^{[l]}, \beta^{[l]} \in [0,1]$,
- **threshold**: $\vartheta^{[l]} \in (0, \infty)$,

where $H := \mathbb{1}_{[0,\infty)}$ is a step function and $T \in \mathbb{N}$ is the number of simulated time steps.

We further define recurrent d.t. LIF-SNN and the function the network realizes:

Definition 2.2. A **recurrent discrete-time LIF-SNN** of **depth** L with **layer-widths** $(n_0, ..., n_{L+1})$ and $T \in \mathbb{N}$ time-steps is given by

$$\Phi := ((W^{[l]}, b^{[l]}, V^{[l]}, u^{[l]}(0), i^{[l]}(0), \alpha^{[l]}, \beta^{[l]}, \vartheta^{[l]})_{l \in [L]}, T, (E, D))$$

where the **input encoder** $E : \mathbb{R}^{n_0} \to \mathbb{R}^{n_0 \times T}$ maps a vector $x \in \mathbb{R}^{n_0}$ to a corresponding first layer spike activation $s^{[0]} = E(x)$ and the **output decoder** $D : \{0,1\}^{n_L \times T} \to \mathbb{R}^{n_{L+1}}$ maps the spike activations of the last hidden layer to real values.

Definition 2.3. A recurrent discrete-time LIF-SNN ϕ **realizes** the function $R(\Phi) : \mathbb{R}^{n_0} \to \mathbb{R}^{n_{L+1}}$:

$$R(\Phi)(x) = D((s^{[L]}(t))_{t \in [T]}) \quad \text{with } s^{[0]} := E(x)$$

Definition 2.4. A recurrent discrete-time LIF-SNN employs **direct encoding** if we have

$$\forall_{t \in [T]} E(x)(t) = x$$

for the input encoder and has **membrane potential outputs** if the output decoder can be written as

$$D((s(t))_{t \in [T]}) = \sum_{t=1}^{T} a_t (W^{[L+1]} s(t) + b^{[L+1]})$$

for some $(a_t)_{t \in [T]} \in \mathbb{R}^T$, $b^{[L+1]} \in \mathbb{R}^{n_{L+1}}$ and $W^{[L+1]} \in \mathbb{R}^{n_{L+1} \times n_L}$.

We will only consider recurrent discrete-time LIF-SNN with direct encoding and membrane potential outputs. In fact, we will use "recurrent discrete-time LIF-SNN" to mean "r. d.t. LIF-SNN with direct encoding and membrane potential".

For clearer construction of our networks we will additionally define neurons:

Definition 2.5. The i-th neuron of a hidden layer $\lambda = (W^{[l]}, b^{[l]}, V^{[l]}, u^{[l]}(0), i^{[l]}(0), \alpha^{[l]}, \beta^{[l]}, \vartheta^{[l]})$ of a r. d.t. LIF-SNN is a tuple (w, b, v, u_0, i_0) with $w \in \mathbb{R}^{n_{l-1}}$, $v \in \mathbb{R}^{n_l}$ and $b, u_0, i_0 \in \mathbb{R}$, such that b, u_0, i_0 are the i-th component of $b^{[l]}$, $u^{[l]}(0)$, $i^{[l]}(0)$ respectively and w, v are the i-th row vector of $W^{[l]}$, $V^{[l]}$ respectively.

Notation 2.1. We will use $[\![x, y]\!]$, $[\![x, y)\!]$ with $x, y \in \mathbb{R}^n$ to write half-open cuboids $\prod_{i=1}^n [x_i, y_i)$, and closed cuboids $\prod_{i=1}^n [x_i, y_i]$ respectively. We further take $[x, y)$ to be empty for $x, y \in \mathbb{R}$ with $x \geq y$, and $[-\infty, x)]$ to mean $(-\infty, x)$.

The following lemmas will be very helpful for the proofs in the following sections:

Lemma 2.1. *We define the following non-recursive formulas with explicit reference to the previous spikes. Let $1 \leq t \leq T$ and $(s_p^{[l]})_{l \in \{0 \dots L'\}}$ such that $s_p^{[l]} : \{0,1\}^{n_l \times t'_l}$ for $l \in \{0 \dots L'\}$. L' and $\forall_l t'_l$ have to be chosen such that the following terms are well-defined, e.g. $L' = L$ and $\forall_l t'_l = T$*

$$i^{[l]}(t; (s_p^{[l]})_l) := (\alpha^{[l]})^t i^{[l]}(0) + \sum_{k=1}^{t} (\alpha^{[l]})^{t-k} \left(W^{[l]} s_p^{[l-1]}(k) + V^{[l]} s_p^{[l]}(k-1) \right), \tag{5}$$

$$p^{[l]}(t; (s_p^{[l]})_l) := (\beta^{[l]})^t u^{[l]}(0) + \sum_{k=1}^{t} (\beta^{[l]})^{t-k} \left(i^{[l]}(k; (s_p^{[l]})_l) + b^{[l]} \right) - \vartheta \sum_{k=1}^{t-1} (\beta^{[l]})^{t-k} s_p^{[l]}(k), \tag{6}$$

$$s^{[l]}(t; (s_p^{[l]})_l) := H(p^{[l]}(t; (s_p^{[l]})_l) - \vartheta 1_{n_l}) \tag{7}$$

$$u^{[l]}(t; (s_p^{[l]})_l) := (\beta^{[l]})^t u^{[l]}(0) + \sum_{k=1}^{t} (\beta^{[l]})^{t-k} \left(i^{[l]}(k; s_p) + b^{[l]} - \vartheta s_p^{[l]}(k) \right), \tag{8}$$

We use the convention $\forall_{\alpha, \beta \in [0,1]} \alpha^0, \beta^0 = 1$. These formulas are equivalent to the previous recursive definitions, given $s_p^{[l]} = s^{[l]}$ for $l \in \{0 \dots L'\}$.

Proof. Let the time $1 \leq t \leq T$ be non-zero. By induction we immediately get

$$i^{[l]}(t) = (\alpha^{[l]})^t i^{[l]}(0) + \sum_{i=1}^{t} (\alpha^{[l]})^{t-i} \left(W^{[l]} s^{[l-1]}(i) + V^{[l]} s^{[l]}(i-1) \right)$$

By definition of $u^{[l]}$ and $p^{[l]}$ we further obtain

$$u^{[l]}(t) = \beta^{[l]} u^{[l]}(t-1) + i^{[l]}(t) + b^{[l]} - \vartheta s^{[l]}(t)$$

from which

$$u^{[l]}(t) = (\beta^{[l]})^t u^{[l]}(0) + \sum_{i=1}^{t} (\beta^{[l]})^{t-i} \left(i^{[l]}(i) + b^{[l]} - \vartheta s^{[l]}(i) \right)$$

follows by induction. By substituting $u^{[l]}$ in $p^{[l]}$ we further obtain:

$$p^{[l]}(t) = (\beta^{[l]})^t u^{[l]}(0) + \sum_{i=1}^{t} (\beta^{[l]})^{t-i} \left(i^{[l]}(i) + b^{[l]} \right) - \vartheta \sum_{i=1}^{t-1} (\beta^{[l]})^{t-i} s^{[l]}(i)$$

□

Lemma 2.2. *Let $t_0, t_\omega \in [T]$ and $j \in [n_l]$ for an $l \in [L]$ such that $t_0 \leq t_\omega$. If $\forall_{t\in\{t_0 \dots t_\omega\}} i_j^{[l]}(t) + b_j^{[l]} \leq \vartheta$ and $u_j^{[l]}(t_0 - 1) < \vartheta$, then $\forall_{t\in\{t_0 \dots t_\omega\}} u_j^{[l]}(t) < \vartheta$.*

If further $\beta = 1$ and $u_j^{[l]}(t_0 - 1) + \sum_{t=t_0}^{t_\omega} (i_j^{[l]}(t) + b_j^{[l]}) \geq 0$

$$\left(u_j^{[l]}(t_0 - 1) + \sum_{t=t_0}^{t_\omega} (i_j^{[l]}(t) + b_j^{[l]}) \right) - \vartheta \sum_{t=t_0}^{t_\omega} s_j^{[l]}(t) < \vartheta$$

If either $\forall_{t\in\{t_0 \dots t_\omega\}} s_j^{[l]}(t) = 0$ or $\forall_{t\in\{t' \dots t_\omega\}} i_j^{[l]}(t) + b_j^{[l]} \geq 0$, where t' is the latest time $\leq t_\omega$ such that $s_j^{[l]}(t') = 0$, then we even get

$$0 \leq \left(u_j^{[l]}(t_0 - 1) + \sum_{t=t_0}^{t_\omega} (i_j^{[l]}(t) + b_j^{[l]}) \right) - \vartheta \sum_{t=t_0}^{t_\omega} s_j^{[l]}(t)$$

Remark 2.1. If $\vartheta = 1$, we get in particular

$$\left\lfloor u_j^{[l]}(t_0 - 1) + \sum_{t=t_0}^{t_\omega} (i_j^{[l]}(t) + b_j^{[l]}) \right\rfloor = \sum_{t=t_0}^{t_\omega} s_j^{[l]}(t)$$

Proof. We proof the first part by induction: Let $t \in \{t_0 \dots t_\omega\}$. By definition of $p^{[l]}$, by the given assumptions and by induction hypothesis we have

$$p_j^{[l]}(t) = \beta^{[l]} u_j^{[l]}(t-1) + i_j^{[l]}(t) + b_j^{[l]} \leq u_j^{[l]}(t-1) + i_j^{[l]}(t) + b_j^{[l]} < 2\vartheta$$

By definition of $u^{[l]}$ and $s^{[l]}$, we further get $u_j^{[l]}(t) < \vartheta$.

Let us further assume $\beta = 1$ and $u_j^{[l]}(t_0 - 1) + \sum_{t=t_0}^{t_\omega} (i_j^{[l]}(t) + b_j^{[l]}) \geq 0$. We now get

$$u^{[l]}(t_\omega) = u^{[l]}(t_0 - 1) + \sum_{k=t_0}^{t_\omega} \left(i^{[l]}(k) + b^{[l]} - \vartheta s^{[l]}(k) \right),$$

due to Lemma 2.1. From which we immediately obtain

$$\left(u_j^{[l]}(t_0 - 1) + \sum_{t=t_0}^{t_\omega} (i_j^{[l]}(t) + b_j^{[l]}) \right) - \vartheta \sum_{t=t_0}^{t_\omega} s_j^{[l]}(t) < \vartheta$$

by the first part of the proof. Let us further proof the left part of the inequality: Let t' be the latest time that $s_j^{[l]}(t') = 1$ has held for $t' \in \{t_0 \dots t_\omega\}$. We may assume that t' exists since we are otherwise immediately finished. We then get $\vartheta \leq p_j^{[l]}(t')$ and therefore

$$0 \leq u_j^{[l]}(t') = u_j^{[l]}(t_0 - 1) + \sum_{k=t_0}^{t'} \left(i_j^{[l]}(k) + b_j^{[l]} - \vartheta s_j^{[l]}(k) \right)$$

by Lemma 2.1. By assumption and choice of t' we can conclude

$$\leq u_j^{[l]}(t_0 - 1) + \sum_{k=t_0}^{t_\omega} \left(i_j^{[l]}(k) + b_j^{[l]} - \vartheta s_j^{[l]}(k) \right)$$

□

Let us now take a look at some simple examples:

3 Structure of computations in r. d.t. LIF-SNN s

This section concern itself with the approximation of continuous functions by d.t. LIF-SNN.

In [Nguyen et al., 2025] the following theorem was proved:

Theorem 3.1. *Let f be a continuous function on a compact set $\Omega \subset \mathbb{R}^{n_0}$. For all $\varepsilon > 0$, there exists a d.t. LIF-SNN Φ with direct encoding, membrane potential output, $L = 2$ and $T = 1$ such that*

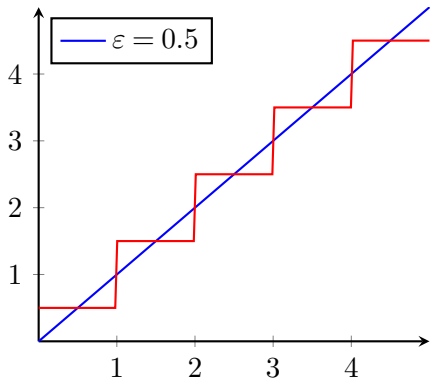
$$\|(R(\Phi) - f)|_\Omega\|_\infty \leq \varepsilon$$

Moreover, if f is Γ-Lipschitz, then Φ can be chosen with width parameter $n = (n_1, n_2)$ given by

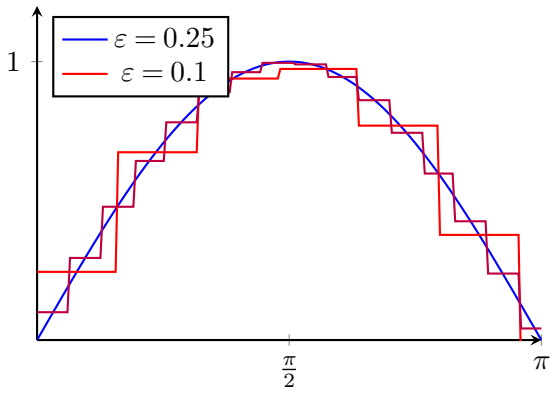
$$n_1 = \left(\max\left\{\left\lceil \frac{\operatorname{diam}_\infty(\Omega)}{\varepsilon}\Gamma \right\rceil, 1\right\} + 1\right) n_0$$
$$n_2 = \max\left\{\left\lceil \frac{\operatorname{diam}_\infty(\Omega)}{\varepsilon}\Gamma \right\rceil^{n_0}, 1\right\}$$

where $\operatorname{diam}_\infty(\Omega) = \sup_{x,y\in\Omega} \|x - y\|_\infty$.

Proof. See [Nguyen et al., 2025]. □



(a) A d.t. LIF-SNN approximating the identity



(b) A d.t. LIF-SNN approximating a sinus wave

The proof of Theorem 3.1 works by first showing that a continuous function can be arbitrarily approximated by step functions, in particular by step functions constant on hypercubes in Ω. Then the d.t. LIF-SNN is constructed by using the first layer to cut the input space by hyperplanes along the cubes and using the second layer to represent the hypercubes.

While quite simple, this construction does not use the unique feature of (r.) d.t. LIF-SNN , the ability of neurons to accumulate state over time. It therefore needs quite a lot more neurons than actually needed for many functions with (almost) linear segments, e.g. a sinus wave, see also Fig. 3.1b.

We will now show a more efficient construction for r. d.t. LIF-SNN that uses the fact that r. d.t. LIF-SNN can quite efficiently approximate linear segments. The general intuition behind is to use piece-wise linear functions to approximate continuously differentiable functions and then approximate those piece-wise linear functions by r. d.t. LIF-SNN . In the end, we will see that we can in fact approximate any continuous function like this, by using mollification to extract an continuously differentiable approximation of the function.

We will now provide an alternative version of this theorem which uses an increased latency to reduce the number of neurons:

Theorem 3.2. *Let $f \in \mathcal{C}^1(U, \mathbb{R}^m)$, with $\emptyset \neq U \subset \mathbb{R}^n$ open, be a continuously differentiable function. Let further $\Omega \subset U$ be a compact set. For all $\varepsilon, \mu, \nu > 0$, $\varepsilon = \mu + \nu$, there exists a r. d.t. LIF-SNN Φ with $L = 3$ and*

$$\begin{aligned} T &= (K(\mu)+1)T_r(\nu)+1 \\ n_1 &= n+1 \\ n_2 &= K^n(\mu)\cdot(n+1)+3 \end{aligned}$$

such that

$$\left\|(R(\Phi)-f)|_\Omega\right\|_{\infty,2} \leq \varepsilon.$$

Here $T_r := T_r(\nu) := \frac{\operatorname{diam}(\Omega)}{K}\frac{\|f'\|_{\infty,2}}{2\nu}$. The definition of $K(\mu)$ is given in Lemma 3.3.

Lemma 3.1. *Let $\Omega \subset \mathbb{R}^n$ be compact. Then there exists a half-open cube C with width $\operatorname{diam}_\infty(\Omega)$ such that $\Omega \subset \overline{C}$.*

Proof of Lemma 3.1.. We can first define $x := (\min_{x\in\Omega} x_i)_i$ since Ω is compact We further have $y := x + \operatorname{diam}_\infty(\Omega)\cdot \mathbb{1}_n$. We now have $C := [\![x, y)\!)$. Suppose now a point $z \in \Omega \setminus \overline{C}$ exists. By definition of x, we have $x \leq z$. By definition of C we further get $z \not\leq y$, and therefore $\exists_i y_i < z_i$ by definition of C. But this means $\|x-z\|_\infty > \operatorname{diam}_\infty(\Omega)$. □

Lemma 3.2. *Let $\omega : [0,\infty] \to [0,\infty]$ be a modulus of uniform continuity of a uniformly continuous function $f : M \to N$, where M, N are metric spaces. We then define the generalized inverse of $\omega : [0,\infty] \to [0,\infty]$ by $\omega^\dagger(s) := \inf\{t \in [0,\infty] \mid \omega(t) > s\}$.*

We have the following properties

1. $\forall_{x,y\in M, s\in[0,\infty]} d_M(x,y) \leq \omega^\dagger(s) \Rightarrow d_N(f(x),f(y)) \leq s$.
2. $\forall_{s\in[0,\infty]} s = 0 \Leftrightarrow w^\dagger(s) = 0$.

Proof.

1. Let $x, y \in M$ and $s \in [0,\infty]$ be given such that $d_M(x,y) \leq \omega^\dagger(s)$. By definition of $\omega^\dagger$, this means $\omega(d_M(x,y)) \leq s$. Since ω is a modulus of uniform continuity of f, we have $d_N(f(x),f(y)) \leq \omega(d_M(x,y))$ and therefore overall $d_N(f(x),f(y)) \leq s$.
2. Since ω is a modulus of uniform continuity, it is by definition continuous at 0. Let us choose an arbitrary sequence $(t_n)_{n\in\mathbb{N}}$ with $t_n \to 0$. Then $\omega(t_n) \to 0$ and therefore $\omega^\dagger(0) \leq \inf_{n\in\mathbb{N}} t_n = 0$.

   Is on the other hand $\omega^\dagger(s) = 0$, then there is a sequence $(t_n)_{n\in\mathbb{N}}$ with $t_n \to 0$ and therefore $\omega(t_n) \to 0$. By definition of $\omega^\dagger(s)$, we have $\omega(t_n) > s$, so we get $s = 0$.

□

Lemma 3.3. *Let $f \in \mathcal{C}^1(U, \mathbb{R}^m)$, with $\emptyset \neq U \subset \mathbb{R}^n$ open, be a continuously differentiable function. Let further $\Omega \subset U$ be a compact set, such that there is a half-open cube C with $\Omega \subset \overline{C} \subset U$.*

For every $\mu > 0$ we can compose C into

$$K^n := K(\mu)^n := \min_{\substack{\xi,\theta>0 \\ \xi\theta=\mu}} \left\{ \left\lceil \frac{\operatorname{diam}_\infty(\Omega)}{\frac{2}{\sqrt{n}}\min(\omega^\dagger(\xi),\theta)} \right\rceil \right\}^n$$

half-open subcubes $(C_i)_{i=1..K^n}$ *such that affine linear functions* $g_i : C_i \to \mathbb{R}^m$ *exist with* $\|f-g\|_{\infty,2} < \mu$*, where* g *is the continuous extension of* $(\sum_{i=1}^m g_i \mathbb{1}_{C_i})|_C$ *on* $\overline{C}$*. Here* $\omega^\dagger$ *is the generalized inverse of a modulus of uniform continuity (with regard to* $\|\cdot\|_2$*) of the total derivative* $dF|_{\overline{C}}$*. Since* $\xi > 0$*, we also have* $\omega^\dagger(\xi) > 0$ *by Lemma 3.2.*

Remark 3.1. 1. If $f \in \mathcal{C}^1(U, \mathbb{R}^m)$ is given with $\emptyset \neq U \subset \mathbb{R}^n$ and a compact $\Omega \subset \mathbb{R}^n$, but with no half-open cube C such that $\Omega \subset \overline{C} \subset U$, we can extend $f|_\Omega$ to a function $f' \in \mathcal{C}^1(\mathbb{R}^n, \mathbb{R}^m)$: There is a partition of one $\varphi_1, \varphi_2 \in \mathcal{C}^\infty(\mathbb{R}^n)$ subordinate to U and $\mathbb{R}^n \setminus \Omega$. We get $\varphi_1|_\Omega = 1$ and therefore $(\varphi_1 f)|_\Omega = f|_\Omega$. $f' := \varphi_1 f$ is further clearly $\mathcal{C}^1(U, \mathbb{R}^m)$.

Proof of Lemma 3.3.. By Lemma 3.1 we have a half-open cube C with width $\operatorname{diam}_\infty(\Omega)$ and $\Omega \subset \overline{C}$. We can assume w.l.o.g. we can assume $\overline{C} \subset U$, since we can extend f to a function $f' \in \mathcal{C}^1(\mathbb{R}^n, \mathbb{R}^m)$:

Let further $\mu > 0$ be given. Since Ω is compact and $f \in \mathcal{C}^1(\Omega, \mathbb{R}^m)$, dF is uniformly continuous on Ω. Let ω be a modulus of uniform continuity of $dF|_\Omega$. We will now partition C in

$$K^n := \min_{\substack{\xi,\theta>0 \\ \xi\theta=\mu}} \left\{ \left\lceil \frac{\operatorname{diam}_\infty(\Omega)}{\frac{2}{\sqrt{n}} \min(\omega^\dagger(\xi), \theta)} \right\rceil \right\}^n$$

smaller half-open cubes. There are ξ, θ such that the minimum in the definition of K^n is obtained, since we take it over the set of natural numbers. K is further well-defined, since $\forall_{s>0} \omega^\dagger(s) > 0$, The subcubes have width $w := \frac{\operatorname{diam}_\infty(\Omega)}{K} \leq \frac{2}{\sqrt{n}} \min(\omega^\dagger(\xi), \theta)$.

Let us further define $g_i : C_i \to \mathbb{R}^m$ by $g_i(x) := f(c_i) + df_{c_i}(x - c_i)$ where c_i is the center of C_i, so in particular

$$\|x - c_i\|_2 = \sqrt{\sum_{j=1}^n \|(x-c_i)_j e_j\|_2^2} \leq \frac{w}{2}\sqrt{n} \leq \min(\omega^\dagger(\xi), \theta). \tag{9}$$

It suffices now to show $\left\|f|_{C_i} - g_i\right\|_\infty < \mu$. Let $x \in C_i$ and $h(t) := f(t(x-c_i)+c_i)$ with

$$h'(t) = (df_{(x-c_i)t+c_i} \circ d(t \mapsto t(x-c_i)+c_i)_t)(1) = df_{(x-c_i)t+c_i}(x-c_i).$$

We obtain by the Fundamental theorem of calculus:

$$\begin{aligned}\left\|f(x) - f(c_i) - df_{c_i}(x-c_i)\right\|_2 &= \left\|h(1) - h(0) - df_{c_i}(x-c_i)\right\|_2 \\ &= \left\|\int_0^1 df_{(x-c_i)t+c_i}(x-c_i)dt - df_{c_i}(x-c_i)\right\|_2\end{aligned}$$

Due to the generalized Minkowski-Inequality we can move the norm inside the integral:

$$\begin{aligned}\left\|\int_0^1 df_{(x-c_i)t+c_i}(x-c_i)dt - df_{c_i}(x-c_i)\right\|_2 &\leq \int_0^1 \left\|df_{(x-c_i)t+c_i}(x-c_i) - df_{c_i}(x-c_i)\right\|_2 dt \\ &= \int_0^1 \left\|(df_{(x-c_i)t+c_i} - df_{c_i})(x-c_i)\right\|_2 dt \\ &\leq \int_0^1 \left\|df_{(x-c_i)t+c_i} - df_{c_i}\right\| \|x-c_i\|_2 dt \\ &\leq \int_0^1 \xi \|x-c_i\|_2 dt \\ &= \xi \|x-c_i\|_2 \\ &\leq \xi\theta \\ &= \mu\end{aligned}$$

In the fourth step we use $\left\|df_{(x-c_i)t+c_i} - df_{c_i}\right\| \leq \xi$, which holds due to $\forall_{t\in[0,1]}(x-c_i)t+c_i \in C_i$, (9) and Lemma 3.2. □

Proof of Theorem 3.2.. Let a continuously differentiable function $f \in \mathcal{C}^1(\Omega, \mathbb{R}^m)$ on a compact set $\Omega \subset \mathbb{R}^n$ be given. Let there further be $\varepsilon, \mu, \nu > 0$ with $\varepsilon = \mu + \nu$. By Lemma 3.3 we have a half-open cube $C = [\![x^C, y^C)\!)$ with $\Omega \subset C$ with composition K^n half-open subcubes $(C_i)_{i=1..K^n}$ and linear functions $g_i : C_i \to \mathbb{R}^m$, such that $\|f - g|_\Omega\|_\infty < \mu$ for a $g := \sum_{i=1}^m g_i \mathbb{1}_{C_i}$.

We will now define a r. d.t. LIF-SNN Φ with direct input encoding and membrane-potential outputs such that $\|R(\Phi)|_\Omega - g\|_\infty < \nu$.

Before anything else we shall set the following basic parameters $i^{[l]}(0) = 0$, $\alpha^{[l]} = 0$ and $\beta^{[l]} = \vartheta^{[l]} = 1$ for all layers.

The intuitive idea for the construction of the network is the following: In the first layer we have n neurons which capture the position of the input vector regarding C in their respective dimension and one last neuron that acts as an "alarm clock" that shuts down the other neurons of the layer after the first KT_r time steps.

The general idea of the second layer is that it just listens for the first KT_r time steps to the first layer and then in time-step $KT_r + 1$ it is decided which region C_i the input is located in. The last T_r time-steps are used to encode the location of x inside of C_i.

For each region C_i we have $n+1$-neurons. Each of the first n neurons encodes a commponent of the linear part of g_i. They are also used to inform the $n+1$-th neuron of the group if the x has at least as big as the base point of C_i. The $n+1$-th deactivates all other neurons of regions with smaller base point and encodes the constant part of g_i. The last 3 neurons act as "clock neurons" enabling and disabling the other ones.

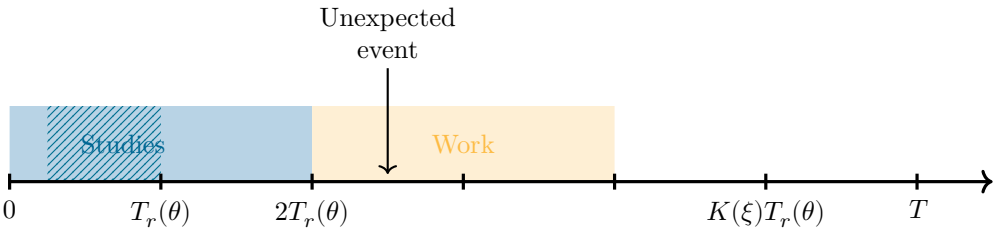


Figure 3.2: Timeline of first-layer neuron

1. First layer: We define the i-th neuron of the n neurons of the first layer by parameters

$$w = \frac{1}{y_i^C - x_i^C} e_i, \quad b = -\frac{x_i^C}{y_i^C - x_i^C}, \quad v = -e_{n+1}, \quad u_0 = 0, \quad i_0 = 0.$$

   The "clock neuron" of the first layer, with index $c_1 := n+1$, is defined by:

$$w = 0, \quad b = \frac{1}{KT_r}, \quad v = e_{c_1}, \quad u_0 = 0, \quad i_0 = 0.$$

2. Second layer: Let us now construct the second layer in the following way: For each of the $K^n(\mu)$ subcubes in C we define $n+1$ neurons like so: Let $C_j = [\![x^{C_j}, y^{C_j})\!)$ be one such subcube with position $q \in \{0, ..., K(\mu)-1\}^{n_1}$ in C. We will write $\iota_j(i) := j(n+1)+i$ to index the first n neurons in the layer and $\omega_j := (j+1)(n+1)$ to index the last neuron of each group.

The i-th neuron of the first n neurons, with index $\iota_j(i)$ in the second layer, has the parameters

$$w = e_i, \quad b = 0, \quad v = T(e_{c_2} - 2e_{c_3} + e_{\omega_j} - r(q)),$$
$$u_0 = -q_i T_r - T + 1, \quad i_0 = 0.$$

where $j(q)$ is the index of the subcube at position q and "the switch" is

$$r(q) := e_{\omega_j} - \sum_{\substack{q' \in \{0,\dots,K(\mu)-1\}^{n_1} \\ q' < q}} e_{\omega_j(q)}.$$

The final neuron of the group, with index ω_j in its layer, has the parameters

$$w = 0, \quad b = 0, \quad v = \frac{1}{n}\sum_{i=1}^{n} e_{\iota_j(i)} + e_{\omega_j} - e_{c_4} - r(q), \quad u_0 = 0, \quad i_0 = 0.$$

We also define the three "clock neurons", with index $c_2 := (j+1)K^n(\mu)+1$, $c_3 := (j+1)K^n(\mu)+2$ and $c_4 := (j+1)K^n(\mu)+3$ with very parameters:

$$w = 0, \quad b = b_{c_i}, \quad v = -2e_{c_i}, \quad u_0 = 0, \quad i_0 = 0.$$

where $b_{c_1} = \frac{1}{KT_r - 1}$, $b_{c_2} = \frac{1}{KT_r}$ and $b_{c_3} = \frac{1}{KT_r}$.

3. Output decoder: We further define the parameters of the output decoder by $a_t = 0$, for $t \leq KT_r + 1$ and otherwise $a_t = 1$. We further set $b^{[L+1]} = 0$ and $(W^{[L+1]})_{l,\iota_j(i)} = g(x^{C_j} + \frac{1}{T_r}e_i) - g(x^{C_j})$ for $l \in [m]$, $j \in K^n$ and $i \in [n]$, as well as $(W^{[L+1]})_{l,\omega_j} = g(x^{C_j})$ for $l \in [m]$ and $j \in K^n$.

We will now proof that this construction indeed approximates g well enough. It will be helpful to consider the following, by choice of $i^{[l]}, \alpha^{[l]}, \beta^{[l]}, \vartheta^{[l]}$, simplified equations:

$$p^{[l]}(t) = u^{[l]}(t-1) + W^{[l]}s^{[l-1]}(t) + V^{[l]}s^{[l]}(t-1) + b^{[l]}$$
$$s^{[l]}(t) = H(p^{[l]}(t) - 1_{n_l})$$
$$u^{[l]}(t) = p^{[l]}(t) - s^{[l]}(t)$$

and in particular by Lemma 2.1

$$i^{[l]}(t) = W^{[l]}s^{[l-1]}(t) + V^{[l]}s^{[l]}(t-1)$$
$$p^{[l]}(t) = u^{[l]}(0) + \sum_{k=1}^{t}\left(W^{[l]}s^{[l-1]}(k) + V^{[l]}s^{[l]}(k-1) + b^{[l]}\right) - \sum_{k=1}^{t-1} s^{[l]}(k).$$

Let now $s^{[0]}(t) = x \in C$. We will proof $\|R(\Phi)(x) - g(x)\|_{\infty,2} \leq \nu$ in steps, by first proofing some simple properties of the previously defined neurons:

1. Characterization of the "clock neuron" of the first layer:
   Let us first regard the neuron in the first layer: By choice of parameters we get:

$$p^{[1]}_{c_1}(t) = \frac{t}{KT_r} + \sum_{k=1}^{t} s^{[1]}_{c_1}(k-1) - \sum_{k=1}^{t-1} s^{[1]}_{c_1}(k) = \frac{t}{KT_r}$$

   So $s^{[1]}_{c_1}(t) = 1 \Leftrightarrow t \geq KT_r$.

2. Characterization of i-th neuron of the first layer, "capturing" the i-th dimension:
We have:

$$i_i^{[1]}(t)+b_i^{[1]}=\frac{x_i-x_i^C}{y_i^C-x_i^C}-s_{c_1}^{[1]}(t-1)$$

So we can use Lemma 2.2 for $x_i^C \le x_i < y_i^C$ and $t \le KT_r$ to obtain

$$\left\lfloor KT_r\frac{x_i-x_i^C}{y_i^C-x_i^C}\right\rfloor=\left\lfloor u_i^{[1]}(0)+\sum_{t=1}^{KT_r}(i_i^{[1]}(t)+b_i^{[1]})\right\rfloor=\sum_{t=1}^{KT_r}s_i^{[1]}(t)$$

and $u_i^{[1]}(KT_r)<1$. We further clearly have $i_i^{[1]}(t)+b_i^{[1]}<0$ for $x_i^C \le x_i < y_i^C$ and therefore

$$p_i^{[1]}(t)=u_i^{[1]}(KT_r)+\sum_{k=KT_r+1}^{t}\left(i_i^{[1]}(t)+b_i^{[1]}\right)-\sum_{k=KT_r+1}^{t-1}s^{[1]}(k)<1$$

for $t>KT_r$. So in particular

$$\left\lfloor KT_r\frac{x_i-x_i^C}{y_i^C-x_i^C}\right\rfloor=\sum_{t=1}^{KT_r}s_i^{[1]}(t)=\sum_{t=1}^{T}s_i^{[1]}(t) \tag{10}$$

3. Characterization of the "clock neurons" of the second layer:
In contrast to the clock neuron of the first layer, these neurons only fire once:

$$p_{c_i}^{[2]}(t)=tb_{c_i}-2\sum_{k=1}^{t}s_{c_i}^{[2]}(k-1)-\sum_{k=1}^{t-1}s_{c_i}^{[2]}(k)=tb_{c_i}-3\sum_{k=1}^{t-1}s_{c_i}^{[2]}(k)$$

Let us first consider c_2: We clearly have $p_{c_2}^{[2]}(t)<1$ for $t<KT_r-1$, but $p_{c_2}^{[2]}(KT_r-1)=1$. Further $t<3(KT_r-1)$ for all $t\le T$, so $p_{c_2}^{[2]}(t)<1$ for $t>KT_r-1$. So $\forall_{t\in[T]}s_{c_2}^{[2]}(t)=1_{\{KT_r-1\}}(t)$.

Similarly we obtain $\forall_{t\in[T]}s_{c_3}^{[2]}(t)=\chi_{\{KT_r\}}(t)$ and $\forall_{t\in[T]}s_{c_4}^{[2]}(t)=\chi_{\{KT_r+1\}}(t)$.

4. "Non-activator neurons" of the groups in the second layer don't fire before $t=KT_r$.

$$p_{\iota_j(i)}^{[2]}(t)=-q_iT_r-T+1+\sum_{k=1}^{t}\left(s_i^{[1]}(k)+\langle v_i,s^{[2]}(k-1)\rangle\right)-\sum_{k=1}^{t-1}s^{[2]}(k).$$
$$\langle v_i,s^{[2]}(k-1)\rangle=T(s_{c_2}^{[2]}(k-1)-2s_{c_3}^{[2]}(k-1)+s_{\omega_j}^{[2]}(k-1)-\langle r(q),s^{[2]}(k-1)\rangle)$$

with .

5. "Activator neurons" of the second layer at most once for $t\le K(\mu)T_r(\nu)$/"Activator neurons" activate at earliest at $t=K(\mu)T_r(\nu)+1$:
We get

6. "Activator neurons" of the second layer don't fire before $t=KT_r+1$.

7. "Activator neurons" of the second layer fire at most once at $t=KT_r+1$ and $t=KT_r+2$.

8. "Activator neurons" of the second layer fire once if x is bigger than the base point of their group and twice if x is in the region.

9. "Non-activator neurons" of the groups in the second layer fire only fire after KT_r+1, if x is the corresponding region.

$$p^{[2]}_{\iota_j(i)}(t) = 1 - q_i T_r + \sum_{k=1}^{t} \left(s^{[1]}_i(k) + T\langle r(q), s^{[2]}(k-1)\rangle \right) - (T+1) \sum_{k=1}^{t-1} s^{[2]}_{\iota_j(i)}(k)$$

$$p^{[2]}_{\omega_j}(t) = \sum_{k=1}^{t} \left(\frac{1}{n+1} \left(s^{[1]}_{c_1}(t) + \sum_{i=1}^{n} s^{[2]}_{\iota_j(i)}(t-1) \right) + KT_r \langle r(q), s^{[2]}(k-1)\rangle \right)$$

Notice first, that

for $t \leq K(\mu)T_r(\nu)$, since as we will later show, $s^{[1]}_{\omega_j}(t) = 0$ for all $j \in [K^n]$ and $t \leq K(\mu)T_r(\nu)$. We therefore have $s^{[2]}_{\iota_j(i)}(t) = 1$ for a $t \leq K(\mu)T_r(\nu)$ exactly if $\sum_{k=1}^{K(\mu)T_r(\nu)} s^{[1]}_i(k) \geq q_i T_r(\nu)$ which is equivalent to by (10)

10. "Activator neuron" activates exactly if x is as large as the lower vertex:

11. The neurons of a subcube C_j fire after $K(\mu)T_r(\nu)+1$, exactly if $x \in C_j$

12. If $x \in C_j$, then $\|R(\Phi)(x) - g(x)\|_2 \leq \nu$

13. Characterization of the ω_j-th neuron of the second layer, the "activator neuron" of group j:
    We get

Let now $s^{[0]}(t) = x \in \overline{C} \setminus C$.

□

Sadly the size of the network in this construction is not always smaller than the one from Theorem 3.1. A concrete counter example is a sinus wave with high frequency and small ampitude, like $f(x) := \frac{\sin(nx)}{n}$ with $n \in \mathbb{N}$ on $\Omega = [0, 2\pi]$. Since $f'(x) = \cos(nx)$ and $\|f'|_\Omega\|_\infty = 1$, we can choose $L = 1$ as a Lipschitz constant for f. At the same time, since $f''(x) = n\sin(nx)$ and $\|f''|_\Omega\|_\infty = n$, the biggest possible modulus of uniform continuity on Ω we can give for f' is $\delta(\varepsilon) := \frac{\varepsilon}{n}$. So we get

$$\max_{\substack{\xi,\theta>0 \\ \xi\theta=\varepsilon}} \min(\delta(\xi), \theta) = \max_{\xi>0} \min(\frac{\xi}{n}, \frac{\varepsilon}{\xi}) = \sqrt{\frac{\varepsilon}{n}}$$

So we get that $K(\varepsilon)$ has a value of $\lfloor \frac{\pi\sqrt{n}}{\sqrt{\varepsilon}} \rfloor$ by definition. We therefore get for the layer sizes:

Theorem 3.1	*Theorem* 3.2
$n_1 = \lceil \frac{2\pi}{\varepsilon} \rceil + 1$	$n_1 = 2$
$n_2 = \lceil \frac{2\pi}{\varepsilon} \rceil$	$n_2 = 2\lfloor \frac{\pi\sqrt{n}}{\sqrt{\varepsilon}} \rfloor$

While the first and second layer are clearly far smaller than the first layer of the other construction, especially for small ε, the third layer of our construction is arbitrarily bad for $n \to \infty$ compared to the last layer of the other construction.

But there is one big difference. Since the size of last layer grows only proportionally to $\frac{1}{\sqrt{\varepsilon}}$ and not proportionally to $\frac{1}{\varepsilon}$, it is arbitrarily smaller than the other constructions last layer for $\varepsilon \to 0$ and any $n \in \mathbb{N}$.

We will generalize this observation with the following theorem:

Theorem 3.3. *Let $f \in \mathcal{C}^1(U, \mathbb{R})$, with $\emptyset \neq U \subset \mathbb{R}^n$ open, be a continuously differentiable function, such that $dF|_\Omega$ is L-Lipschitz. Let further $\Omega \subset U$ be a compact set.*

We can choose parameters $(\mu_\varepsilon)_{\varepsilon>0}$, $\mu_\varepsilon := \sqrt{\varepsilon}$, and a modulus of continuity $\omega(x) := Lx$ of $dF|_\Omega$, such that

$$\lim_{\varepsilon \to 0} \left(\frac{K(\mu_\varepsilon)}{\varepsilon^{-1}} \right) = 0$$

where K as defined in Lemma 3.3 for Ω.

Proof. First $\omega(x) := L(x)$ is obviously indeed a modulus of continuity of $dF|_\Omega$, since by definition of L, we have $\forall_{x,y \in \Omega} \|dF_x - dF_y\| \leq L\|x-y\|_2 = \omega(\|x-y\|_2)$ and ω is clearly continuous at 0. We further get $\omega^\dagger(s) = \inf\{t \in [0,\infty] \mid Lt > s\} = \frac{s}{L}$.

It further suffices to show there exist $(\xi_{\mu_\varepsilon})_{\varepsilon>0}$ and $(\theta_{\mu_\varepsilon})_{\varepsilon>0}$ with $\forall_{\varepsilon>0} \xi_{\mu_\varepsilon} \theta_{\mu_\varepsilon} = \mu_\varepsilon$ and

$$\lim_{\varepsilon \to 0} \frac{\varepsilon}{\min(\omega^\dagger(\xi_{\mu_\varepsilon}), \theta_{\mu_\varepsilon})} = 0$$

But this is obvious, if we choose $\varepsilon_{\mu_\varepsilon} := \sqrt{\mu_\varepsilon}$ and $\theta_{\mu_\varepsilon} := \sqrt{\mu_\varepsilon}$, since

$$\frac{\varepsilon}{\min(\omega^\dagger(\xi_{\mu_\varepsilon}), \theta_{\mu_\varepsilon})} = \frac{\varepsilon}{\min(\frac{1}{L}, 1)\sqrt{\mu_\varepsilon}} = \max(L, 1)\varepsilon^{\frac{3}{4}}.$$

□

4 Complexity of input partitions

In the following we will analyze what shape the graph of a r. d.t. LIF-SNN has. Since the output of the following layers only depends on the spike trains of the first hidden layer, those layers are only able to merge different output regions of the first hidden layer. We will therefore only study the output landscape of the first hidden layer.

We can therefore simplify the notation in this section by writing $W, b, V, \beta, \vartheta, u, s$ for $W^{[1]}, b^{[1]}, V^{[1]}, \beta^{[1]}, \vartheta^{[1]}, u^{[1]}, s^{[1]}$ respectively. Since we are using direct encoding, we will write x for $s^{[0]} \in \mathbb{R}^{n_0}$:

$$i(t) = \alpha i(t-1) + Wx + Vs(t-1) \tag{11}$$
$$p(t) = \beta u(t-1) + i(t) + b \tag{12}$$
$$s(t) = H(p(t) - \vartheta 1_n) \tag{13}$$
$$u(t) = p(t) - \vartheta s(t) \tag{14}$$

By using the simplified notation, we obtain the following explicitely formulated definitions from Lemma 2.1 for $x \in \mathbb{R}^{n_0}$ and $s : \{0,1\}^{n_1 \times t'}$ for t' chosen such that the following equations are well-defined:

$$i(t; x; s_p) = \alpha^t i(0) + \sum_{k=1}^{t} \alpha^{t-k} \left(Wx + Vs_p(k-1)\right), \tag{15}$$
$$u(t; x; s_p) = \beta^t u(0) + \sum_{k=1}^{t} \beta^{t-k} \left(i(k; x; s_p) + b - \vartheta s_p(k)\right), \tag{16}$$
$$p(t; x; s_p) = \beta^t u(0) + \sum_{k=1}^{t} \beta^{t-k} \left(i(k; x; s_p) + b\right) - \vartheta \sum_{k=1}^{t-1} \beta^{t-k} s_p(k), \tag{17}$$
$$s(t; x; s_p) = H(p(t; x; s_p) - \vartheta 1_{n_1}) \tag{18}$$

Remark 4.1. We will sometimes just write $i(t; x)$ instead of $i(t; x; s_p)$, with which we imply to use s (at x) for s_p. The same holds for u, p, s.

It is furthermore quite obvious that i grows linear in x, given fixed s_p. The same holds therefore for u and p. Further, the interdependence of the different components in $i/u/p/s$ is gone using a constant s_p. We can therefore split e.g. p into functions $p_{i;s_p}$ such that $p(t, x) = p_{1;s_p}(t; x_1) \times ... \times p_{n_1;s_p}(t; x_{n_1})$.

The functions $i_{i;s_p}, u_{i;s_p}, p_{i;s_p}$ are not only linear in x, but also grow monotonically in x since $\alpha, \beta \geq 0$. If $t \geq 1$, then $i_{i;s_p}, u_{i;s_p}, p_{i;s_p}$ grow even strictly monotonically in x.

So we in particular have that p to is growing monotonically (regarding by-component ordering) on fixed s_p. Since H is a monotonically growing function, s is also growing monotonically with fixed s_p.

We will examine the graph by investigating the shape and location of the constant regions of a r. d.t. LIF-SNN :

Definition 4.1. The set of constant regions of a r. d.t. LIF-SNN Φ is defined as the partition

$$C_\Phi := \{R(\Phi)^{-1}(\{y\}) \mid y \in \mathrm{im}(R(\Phi))\}$$

of $\mathbb{R}^{n_0}$. A constant region with spike train $s' \in \{0,1\}^{n_1 \times T}$, of the first layer of a r. d.t. LIF-SNN Φ is defined as

$$C_{s'} := \{x \in \mathbb{R}^{n_0} \mid \forall_{t \in [T]} s(t; x) = s'(t)\}$$

We further notate the set of these regions by $C_{\Phi,1} := \{C_{s'} \mid s' \in \{0,1\}^{n_1 \times T}, C_{s'} \neq \emptyset\}$.

Proposition 4.1. *The constant regions of the first layer of a r. d.t. LIF-SNN Φ with $W = I_{n_1}$ are half-open cuboids. In particular, let $C_{s'} = [\![x^{s'}, y^{s'}]\!) \in C_{\Phi,1}$ be a constant region. Then*

$$x_i^{s'} = \sup_{\substack{t\in[T]\\ s_i'(t)=1}} g_i(t;s') \qquad y_i^{s'} = \inf_{\substack{t\in[T]\\ s_i'(t)=0}} g_i(t;s')$$

where we use the conventions $\inf\emptyset = \infty$ and $\sup\emptyset = -\infty$.

Lemma 4.1. *There is a function $g(t;s_p)$ such that $\forall_{i\in[n_1]} s_i(t;x;s_p) = 1 \Leftrightarrow \langle w_i, x\rangle \geq g_i(t;s_p)$ defined by*

$$g(t;s_p) := -\frac{\sum_{k=1}^{t}\beta^{t-k}\left(\alpha^k i(0) + b + \sum_{l=1}^{k}\alpha^{k-l}Vs_p(l-1)\right) + \beta^t u(0) - \vartheta\left(1+\sum_{k=1}^{t-1}\beta^{t-k}s_p(k)\right)}{\sum_{k=1}^{t}\beta^{t-k}\sum_{l=1}^{k}\alpha^{k-l}}$$

Where w_i denotes the i-th row vector of W.

Proof of Lemma 4.1. We compute for a $i \in [n_1]$ using Lemma 2.1:

$$\begin{aligned}
&H(p_i(t;x;s_p) - \vartheta)\\
&= H\left(\sum_{k=1}^{t}\beta^{t-k}\left(i_i(k;x;s_p) + b_i\right) + \underbrace{\beta^t u_i(0) - \vartheta\left(1+\sum_{k=1}^{t-1}\beta^{t-k}(s_p)_i(k)\right)}_{(*)}\right)\\
&= H\left(\sum_{k=1}^{t}\beta^{t-k}\left(\alpha^k i_i(0) + \sum_{l=1}^{k}\alpha^{k-l}\left(\langle w_i, x\rangle + (Vs_p(l-1))_i\right) + b_i\right) + (*)\right)\\
&= H\left(\sum_{k=1}^{t}\beta^{t-k}\sum_{l=1}^{k}\alpha^{k-l}\langle w_i, x\rangle + \sum_{k=1}^{t}\beta^{t-k}\left(\alpha^k i_i(0) + b_i + \sum_{l=1}^{k}\alpha^{k-l}(Vs_p(l-1))_i\right) + (*)\right)\\
&= H\left(\langle w_i, x\rangle + \frac{\sum_{k=1}^{t}\beta^{t-k}\left(\alpha^k i_i(0) + b_i + \sum_{l=1}^{k}\alpha^{k-l}(Vs_p(l-1))_i\right) + (*)}{\sum_{k=1}^{t}\beta^{t-k}\sum_{l=1}^{k}\alpha^{k-l}}\right)\\
&= H\left(\langle w_i, x\rangle + g_i(t;s_p)\right)
\end{aligned}$$

□

Lemma 4.2. *Let $C_{s'} \in C_{\Phi,1}$, we then have*

$$C_{s'} = \left(\bigcap_{\substack{i\in[n_1],t\in[T]\\ s_i'(t)=0}} \pi_i^{-1}([-\infty, g_i(t;s'))\right) \bigcap \left(\bigcap_{\substack{i\in[n_1],t\in[T]\\ s_i'(t)=1}} \pi_i^{-1}([g_i(t;s'), \infty)\right)$$

Proof of Lemma 4.2. Let $C_{s'} \in C_{\Phi,1}$ be such a region. We then get

$$\begin{aligned}
C_{s'} &= \bigcap_{i\in[n_1],t\in[T]} \{x \mid s_i'(t) = s_i(t;x)\}\\
&= \bigcap_{i\in[n_1],t\in[T]} \{x \mid s_i'(t) = s_i(t;x;s')\}\\
&= \left(\bigcap_{\substack{i\in[n_1],t\in[T]\\ s_i'(t)=0}} \{x \mid s_i(t;x;s') = 0\}\right) \bigcap \left(\bigcap_{\substack{i\in[n_1],t\in[T]\\ s_i'(t)=1}} \{x \mid s_i(t;x;s') = 1\}\right)\\
&= \left(\bigcap_{\substack{i\in[n_1],t\in[T]\\ s_i'(t)=0}} \pi_i^{-1}([-\infty, g_i(t;s'))\right) \bigcap \left(\bigcap_{\substack{i\in[n_1],t\in[T]\\ s_i'(t)=1}} \pi_i^{-1}([g_i(t;s'), \infty)\right)
\end{aligned}$$

by Lemma 4.1. □

Lemma 4.3. $\bigcap_{j\in J}\prod_{i\in[n]} M_{i,j} = \prod_{i\in[n]}\bigcap_{j\in J} M_{i,j}$ *for an index set* J *and sets* $(M_{i,j})_{i\in[n],j\in J}$.

Proof of Lemma 4.3. For every $x = (x_i)_{i\in[n]} \in M := \prod_{i\in[n]}\bigcup_{j\in J} M_{i,j}$ holds:

$$x \in \bigcap_{j\in J}\prod_{i\in[n]} M_{i,j} \Leftrightarrow \forall_{j\in J} x \in \prod_{i\in[n]} M_{i,j} \Leftrightarrow \forall_{j\in J}\forall_{i\in[n]} x_i \in M_{i,j}$$

On the other hand, we have

$$x \in \prod_{i\in[n]}\bigcap_{j\in I} M_{i,j} \Leftrightarrow \forall_{i\in[n]} x_i \bigcap_{j\in I} M_{i,j} \Leftrightarrow \forall_{i\in[n]}\forall_{j\in J} x_i \in M_{i,j}$$

In total we get

$$\bigcap_{j\in J}\prod_{i\in[n]} M_{i,j} = M \cap \bigcap_{j\in J}\prod_{i\in[n]} M_{i,j} = M \cap \prod_{i\in[n]}\bigcap_{j\in I} M_{i,j} = \prod_{i\in[n]}\bigcap_{j\in I} M_{i,j}$$

□

Lemma 4.4. *The intersection* $C_1 \cap C_2$ *of half-open cuboids* $C_1, C_2 \subset \mathbb{R}^n$ *is a half-open cuboid. In particular, if* $C_i := \prod_{j\in[n]}[c_{i,j}, d_{i,j})$, *then*

$$C_1 \cap C_2 = \prod_{j\in[n]} ([\sup(c_{1,j}, c_{2,j}), \inf(d_{1,j}, d_{2,j})))$$

We use $\inf/\sup$ *instead of* $\min/\max$ *to highlight that the components of the vertices might be* $\pm\infty$.

Proof of Lemma 4.4. Let us first regard $n = 1$. For $c_1, c_2, d_1, d_2 \in \mathbb{R}\cup\{\pm\infty\}$ we get

$$\begin{aligned} & x \in [c_1, d_1) \cap [c_2, d_2) \\ \Leftrightarrow\ & c_1, c_2 \le x < d_1, d_2 \\ \Leftrightarrow\ & x \in [\sup(c_{1,j}, c_{2,j}), \inf(d_{1,j}, d_{2,j})) \end{aligned}$$

for $x \in \mathbb{R}$ and therefore

$$C_1 \cap C_2 = \prod_{j\in[n]} \left([c_{1,j}, d_{1,j}) \cap [c_{2,j}, d_{2,j})\right) = \prod_{j\in[n]} [\sup(c_{1,j}, c_{2,j}), \inf(d_{1,j}, d_{2,j}))$$

by Lemma 4.3. □

Proof of Proposition 4.1. Let $C_{s'} \in C_{\Phi,1}$ be a constant region. We then get

$$\begin{aligned} C_{s'} &= \left(\bigcap_{\substack{i\in[n_1],t\in[T] \\ s'_i(t)=0}} \pi_i^{-1}([-\infty, g_i(t;s'))) \right) \bigcap \left(\bigcap_{\substack{i\in[n_1],t\in[T] \\ s'_i(t)=1}} \pi_i^{-1}([g_i(t;s'), \infty)) \right) \\ &= \prod_{i\in[n]} [\sup_{t\in[T],\ s'_i(t)=1} g_i(t;s'), \inf_{t\in[T],\ s'_i(t)=0} g_i(t;s')) \end{aligned}$$

by Lemma 4.2 and Lemma 4.4, since

$$\pi_i^{-1}([c,d)) = [-\infty,\infty) \times \ldots \times [c,d) \times \ldots \times [-\infty,\infty).$$

So $C_{s'}$ is a finite intersection of half-open cuboids. We can now apply Lemma 4.4 and obtain that $C_{s'}$ is indeed a half-open cuboid. □

We will add an ordering to spike trains based on lexicographical ordering. We define $s' \leq_l s''$ for $s', s'' \in \{0,1\}^{n_1 \times T}$ to mean that either $s' = s''$ or $s'(t) < s''(t)$ holds for the minimal time t such that the spike trains have different values.

Lemma 4.5. *Let us regard a r. d.t. LIF-SNN Φ with $W = I_{n_1}$. Let further $x, y \in \mathbb{R}^{n_0}$. We then have $x \leq y \Rightarrow s(\cdot; x) \leq s(\cdot; y)$.*

Is on the other hand $s(\cdot; x) < s(\cdot; y)$ lexicographicaly, so $x_i < y_i$ with $i \in [n_1]$, such that t is the smallest time with $\exists_{i \in [n_1]} s_i(t; x) \neq s_i(t; y)$.

Proof. Let $x \leq y$ and $s(\cdot; x) \neq s(\cdot; y)$. We then have a minimal $t \in [T]$ such $s(t; x) \neq s(t; y)$. Now due to $\forall_{t' < t} s(t'; x) = s(t'; y)$ and Remark 4.1 we have $\forall_{i \in [n_1]} s_1(t; x_i) \leq s_i(t; y_i)$ and therefore $s(t; x) < s(t; y)$.

Let now $s(\cdot; x) < s(\cdot; y)$. Then there is a smallest time t such that $\exists_{i \in [n_1]} s_i(t; x) \neq s_i(t; y)$. By definition of the ordering on spike trains we get $s_i(t; x) < s_i(t; y)$. Now $s_{i; s_p}(t; x_i)$ is a monotone function in x_i with $s_p(t) \coloneqq s(t; x)$ by choice of t and Remark 4.1. We therefore have $x_i < y_i$ □

Lemma 4.6. *Let $s' \in \{0,1\}^{n_1 \times T}$ such that $\forall_{i \in [n_1]} \forall_{t, t' \in [T]} s'_i(t) = s'_i(t')$. Let further $C_{s'} = [\![x^{s'}, y^{s'}]\!)$ be the corresponding region. Then $C_{s'}$ is non-empty, i.e. $\forall_{i \in [n_1]} x_i^{s'} < y_i^{s'}$, and $x_i^{s'} = -\infty$ exactly if $\forall_{t \in [T]} s'_i(t) = 0$ and $y_i^{s'} = \infty$ exactly if $\forall_{t \in [T]} s'_i(t) = 1$ for all $i \in [n_1]$.*

Furthermore, $C_{s'}$ has exactly one finite vertex.

Proof. By Remark 4.1 $p_{i; s_p}(t; x)$ is a non-constant linear function for $t \geq 1$ and fixed s_p. We have therefore $\exists_z s_{i; s_p}(t; (-\infty, z_i)) = 0$ and $\exists_z s_{i; s_p}(t; [z_i, \infty)) = 1$ for any fixed $t \in [T]$ and s_p. Since there are only finitely many time-steps $t \in [T]$ and spike trains s_p and, we get $s_{i; s_p}(t; (-\infty, z_i)) = 0$ or $s_{i; s_p}(t; [z_i, \infty)) = 1$ respectively for any s_p and $t \in [T]$, as well as any z with z_i small or large enough.

By choosing the component z_i correctly small or large enough we therefore get a $z \in C_{s'}$. We further get for any $i \in [n_1]$ that if $\forall_t s'_i(t) = 0$, then by construction $s_i(t; (-\infty, z_i)) = 1$, so $\forall_{\delta > 0} z - \delta e_i \in C_{s'}$ and therefore $x_i = -\infty$. If on the other hand $\forall_t s'_i(t) = 1$, then $s_i(t; [z_i, \infty)) = 0$, so $\forall_{\delta \geq 0} z + \delta e_i \in C_{s'}$ and therefore $y_i = \infty$.

Since $T \geq 1$ by definition, we have either $x_i^{s'} = -\infty$ or $y_i^{s'} = \infty$ by definition of s' for each $i \in [n_1]$. By choosing the only finite one from $\{x_i^{s'}, y_i^{s'}\}$ for each $i \in [n_1]$ we get the only finite vertex of $C_{s'}$. □

Lemma 4.7. *If $q \in \mathbb{R}^{n_1}$ is a (finite) vertex of a region $C_{s'} = [\![x^{s'}, y^{s'}]\!) \in C_{\Phi, 1}$, $s' \in \{0,1\}^{n_1 \times T}$, then $\forall_{i \in [n_1]} \forall_{\delta > 0} s_i(\cdot; q) \neq s_i(\cdot; q - \delta(\partial_q)_i e_i)$. Here $\partial_q \coloneqq 2H(x^{s'} + y^{s'} - 2q) - 1$ points is a vector pointing from q roughly in the direction of the opposite vertex of the cube $C_{s'}$ (We allow $x^{s'}, y^{s'}$ to have $-\infty$ or ∞ as components here).*

Proof. Proof by contradiction: If the statement is wrong, than there exists an $i \in [n_1]$, such that for every $\delta > 0$ with $s_i(\cdot; q) = s_i(\cdot; q - \delta(\partial_q)_i e_i)$. Since the other components of $p(\cdot; x)$ as well as $s(\cdot; x)$ are only affected by the value of x_i through $s_i(\cdot; x)$ we further get $s(\cdot; q) = s(\cdot; q - \delta(\partial_q)_i e_i)$. So $q - \delta(\partial_q)_i e_i \in C_{s'}$.

On the other hand $x_i^{s'} \leq q_i - \delta(\partial_q)_i < y_i^{s'}$ cannot be true: Assume that it is. Since q is a (finite) vertex of $[\![x^{s'}, y^{s'}]\!)$, we have $q_i \in \{x_i^{s'}, y_i^{s'}\}$. Case distinction: Suppose $q_i = x_i^{s'}$. Then $(\partial_q)_i = 1$ and $x_i^{s'} \leq q_i - \delta(\partial_q)_i$ is wrong. Suppose on the other hand $q_i = y_i^{s'}$. Then $(\partial_q)_i = -1$ and $q_i - \delta(\partial_q)_i < y_i^{s'}$ is wrong. □

Lemma 4.8. *Points $x \in \mathbb{R}^{n_1}$ with only non-constant spiketrains, i.e. $\forall_{i \in [n_1]} \exists_{t, t' \in [T]} s_i(t; x) \neq s_i(t'; x)$ are contained in regions $C_{s'}$ with only finite vertices.*

Proof. □

Proposition 4.2. *Let P be the set of all finite vertices of regions $C_{s'}$, $s' \in \{0,1\}^{n_1 \times T}$, with constant spiketrains, i.e. $\forall_{t,t' \in [T]} s'(t) = s'(t')$. Then all points $x \in \mathbb{R}^{n_1}$ with only non-constant spiketrains, i.e. $\forall_{i \in [n_1]} \exists_{t,t' \in [T]} s_i(t;x) \neq s_i(t';x)$, are contained in $[\![x^C, y^C]\!]$, where*

$$x_i^C := \min_{\substack{t \in [T],\ s_p \in \{0,1\}^{n_1 \times T} \\ \forall_{t \in [T]} (s_p)_i(t) = 0}} g(t; s_p)$$
$$y_i^C := \max_{\substack{t \in [T],\ s_p \in \{0,1\}^{n_1 \times T} \\ \forall_{t \in [T]} (s_p)_i(t) = 1}} g(t; s_p)$$

We get in particular

$$x^C = \min_{t \in [T]} \left(-\frac{\sum_{k=1}^t \beta^{t-k}(\alpha^k i(0) + b - \vartheta \mathbb{1}) + \beta^t u(0)}{\sum_{k=1}^t \beta^{t-k} \sum_{l=1}^k \alpha^{k-l}} - 1_{t \neq 1} \max(V, 0) \mathbb{1}_{n_1} \right)$$
$$y^C = \max_{t \in [T]} \left(-\frac{\sum_{k=1}^t \beta^{t-k}(\alpha^k i(0) + b - \vartheta \mathbb{1}) + \beta^t u(0)}{\sum_{k=1}^t \beta^{t-k} \sum_{l=1}^k \alpha^{k-l}} - 1_{t \neq 1} \min(V, 0) \mathbb{1}_{n_1} \right)$$

Here $\max(V,0)$ *is component-wise application of* $\max(\cdot,0)$ *on* V*. The expression* $\min(V,0)$ *is defined analogously.*

Proof. Let $i \in [n_1]$ and $x \in \mathbb{R}^{n_1}$ be given with $s^x := s(\cdot; x)$ and $\forall_{t,t' \in [T]} s_i^x(t) = s_i^x(t')$. If $s_i^x = 0$, then $x_i < \inf_{t \in [T]} g_i(t; s^x) \leq x_i^C$. We similarly get $x_i^C \leq \sup_{t \in [T]} g_i(t; s^x) \leq x_i$ for $s_i^x = 1$. So $x \notin [\![x^C, y^C]\!]$ if x has a constant spiketrain in one component. □

5 Experimental results

To try to better understand the landscape of the input of a r. d.t. LIF-SNN to proof **??**, we have created a few programs.

5.1 Computing the number of regions

We assume $W = I_n$ yet again, since we the algorithms become much simpler and more efficient. Also once we understand the situation for $W = I_n$, we are hopeful to proof it for arbitrary W.

Let further g in Algorithm 1 be defined as in Lemma 4.1.

Algorithm 1 Recursive Region Computation

function ComputeRegionsStartingWithST(t, st, x^C, y^C)
  if $t > T$ **then**
    return $\{(st, x^C, y^C)\}$
  end if
  $x \leftarrow g(t; st)$
  $subRegions \leftarrow \{[x', y') \mid x' < y',\ \forall_i x'_i \in \{x_i^C, x_i\},\ \forall_i y'_i \in \{y_i^C, x_i\}\}$
  for $C \in subRegions$ **do**
    $newST \leftarrow \text{append}(st, [(\mathbf{1}_{\{x_i \leq x'_i\}})_{i \in [n_1]}])$
    ComputeRegionsStartingWithST($t+1, newST, x', y'$)
  end for
end function
function ComputeRegions()
  ComputeRegionsStartingWithST($1, [(0, ..., 0)], (-\infty, ..., -\infty), (\infty, ..., \infty)$)
end function

Bibliography

References

[Nguyen et al., 2025] Nguyen, D. A., Araya, E., Fono, A., and Kutyniok, G. (2025). Time to spike? understanding the representational power of spiking neural networks in discrete time.