Information flow in plastic neural network

Minh-Toan Nguyen

INTERNSHIP REPORT

Information flow in plastic neural network

Realized by :
Minh-Toan Nguyen
X2016

Under supervision of :
Mr. Alexandre Muzy (I3S)
Mrs. Patricia Reynaud-Bouret (LJAD)
Mr. Bruno Cessac (INRIA)

8/2019

Acknowledgment

I would like to thank my tutor Mr. Alexandre Muzy, Mrs. Patricia Reynaud-Bouret and Mr. Bruno Cessac for their insightful answers to my questions and for helping me discover as much as possible about research and develop the mindset of a researcher. I would like to give special thanks to Mr. Bruno Cessac for his thoughtful comments and corrections on my report.

Introduction

Computational neuroscience is a branch of neuroscience which uses computational approaches, to study the nervous system. Computational approaches include mathematics, statistics, computer simulations. The computational neuroscience roughly divides into two subfields: theoretical neuroscience focuses on building models that can capture the essential features of the neural system at multiple spatial-temporal scales and neural data science focuses on analyzing the neural data generated by simulations or by experiments.

There is a fruitful interaction between experimental and computational neuroscience. Experimental data helps to build better models and models help to explain experimental results and to gain a deeper understanding of neural systems. Growing computational speed and amount of data is making a great impact on the brain research.

Contents

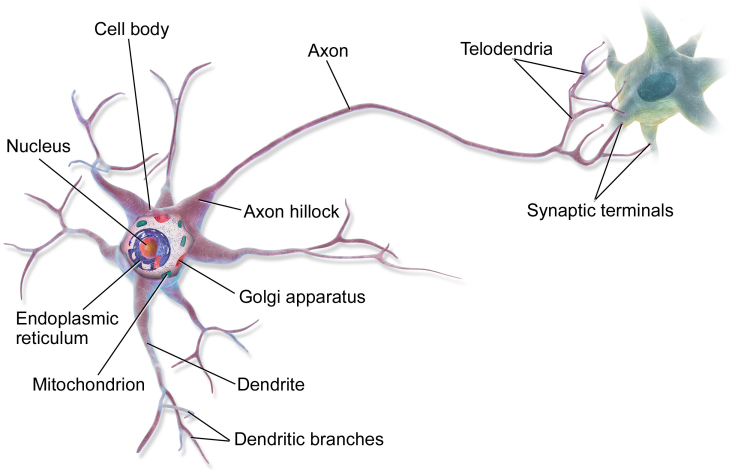

Figure 1: Neuron (*source: Wikipedia*).

1 Basic concepts in neuroscience

1.1 Neuron and action potential

A **neuron** is a specialized cell that is the basic building block of the nervous system. The main components of a neuron are **dendrites**, **cell body** (**soma**), **axon**, and **synaptic terminals** (Figure 1). The electric potential inside a neuron is typically lower than the outside. The **membrane potential** is the difference of these two values, with typical values ranging from –80 mV to 40 mV. This potential is controlled by ion channels on the membrane, which can close or open and let ions travel through it. Neurons are connected to each other and communicate through **spikes** (**action potentials**) which are sudden and short membrane voltage peaks (Figure 2. After the spike there is a If spikes are transmitted from a neuron j to neuron i, then j is called **presynaptic neuron** and i **postsynaptic neuron**. Spikes travel along the axon to synaptic terminals which are connected to dendrites of postsynaptic neurons. The synaptic terminal-dendrite contacts are called **synapses**. When a spike arrives at the synapse, it cause the **synaptic vesicle** to open and release **neurotransmitters**. These transmitters travel through **synaptic cleft** and arrive at the **receptors** on dendrites (Figure 3). All of these happen at a short period of time, causing the membrane potential of postsynaptic neuron to rise (if the presynaptic neuron is **excitatory**) or decay (if presynaptic neuron is **inhibitory**). A neuron is either excitatory or inhibitory: it always excites or always inhibits other neurons but cannot excite one neuron and inhibit another. Inside a network, a neuron is bombarded by spikes from other neurons. Its membrane potential therefore fluctuates and if it crosses a certain threshold, the neuron will spike.

1.2 Learning and synaptic plasticity

When we learn something, at a deeper level, it is the synaptic weights in our brain that change. This process is called **synaptic plasticity**. Synaptic weight (synaptic strength or synaptic efficacy) of the synapse $j \to i$ characterizes the amplitude of the response of neuron i when receiving a spike from neuron j, or the efficacy of transmission from j to i.

In correlation-based models of learning (Hebbian learning), modifications of the synaptic synaptic

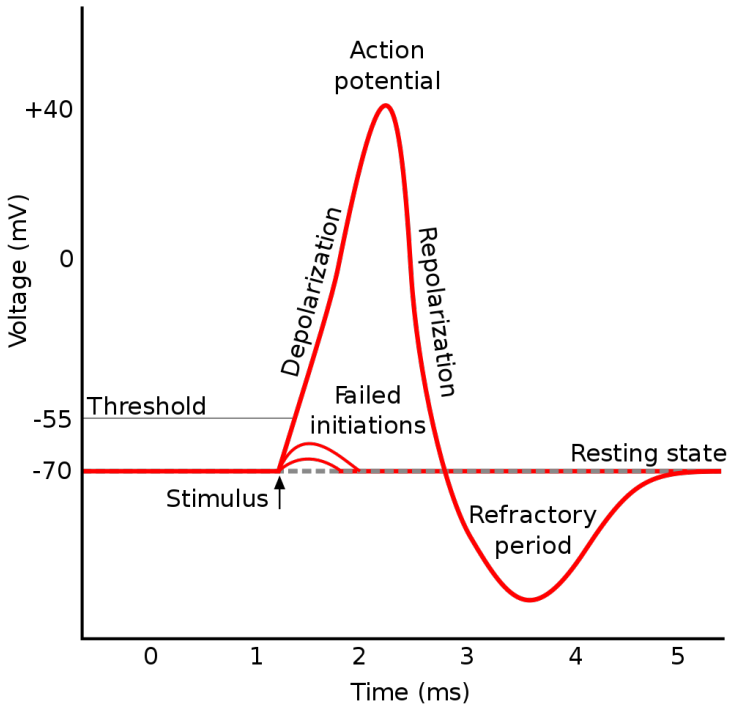

Figure 2: Action potential (*source: Wikipedia*).

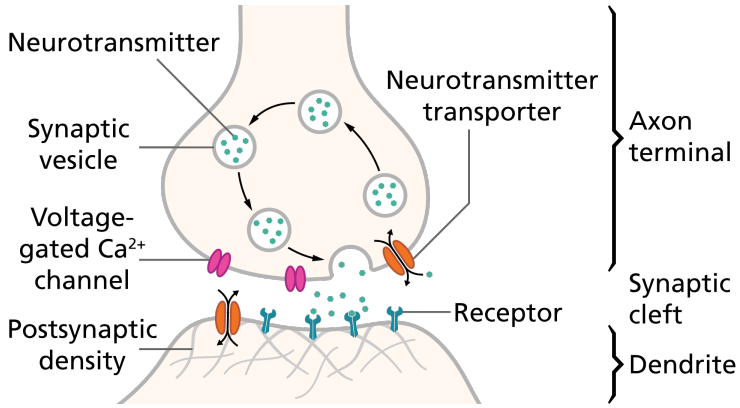

Figure 3: Activities at synapse (*source: Wikipedia*).

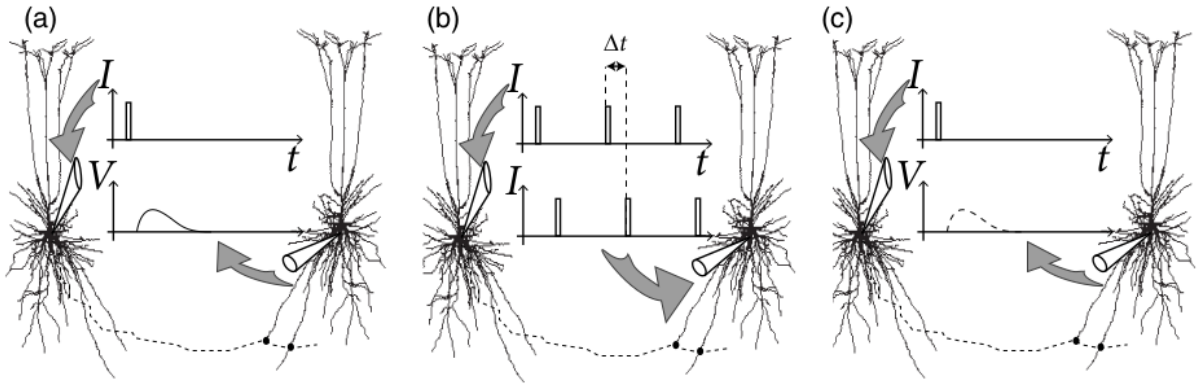

Figure 4: (a) An electrical pulse I makes the presynaptic spike and causes a rise in the membrane potential of postsynaptic neuron. The amplitude of this rise is proportional to the synaptic weight between two neurons. (b) Pairing protocol: both neurons are stimulated with electrical pulses forcing them to spike at precise moments in time. (c) After the pairing protocol, step (a) is repeated to see how the synaptic weight has changed (*source:* [13])

weight is driven by correlations in the firing activity of pre- and postsynaptic neurons. The basic idea is that, neurons that "fire together, wire together".
Spike-timing-dependent plasticity (STDP) takes into account one important factor - time - that was ignored by correlation-based learning. According to STDP, the direction of the synaptic change depends on the relative timing of pre- and postsynaptic spikes. The synapse is strengthened if the presynaptic spike occurs shortly before the postsynaptic spike, but is weakened if the order of spikes is reversed.(Figure 5) This is confirmed by experiments ([6]) in which the pairing protocol is used to measure the effect of STDP for isolated pairs of spike of pre- and postsynaptic neuron. (Figure 4)

2 Objective of the internship

Neurons communicates through spikes. Synaptic weights also change as neurons communicate. It is currently believed that the succession of spikes carries information and there is information that flows between neurons. However, there still not a clear definition of information flow. Loosely speaking, if spiking activities of two neurons are related to each other in some way, we say that there is information that flows between them, in one or both direction. There is still no agreement on how to decipher the information in spike trains as well as how to quantify information flow between neurons. We propose a method to do this via the concept of mutual information. More precisely, we want to measure how the mutual information between neurons in a network evolves under STDP.

In [3], a multivariate Hawkes process is used to model the spiking activity of a neural network. The interactions between neurons are modeled by the interaction functions of Hawkes model. Given the spikes recorded from a small set of neurons, the method Lasso is used to infer the interaction functions of Hawkes model, from which we can reconstruct the connectivity between neurons. Moreover, from these interaction functions we can also estimate the mutual information between neurons. This method of estimating mutual information will be compared with the method in [9] where mutual information are estimated directly from spike trains.

The first part of this internship is to well understand the Lasso method in [3] and test its performance

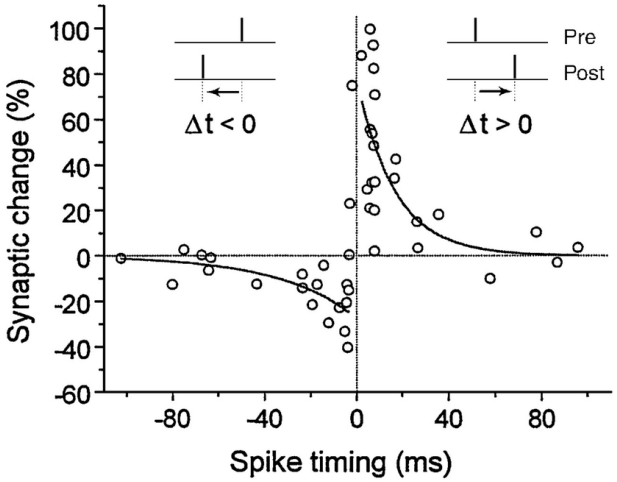

Figure 5: Change of synaptic weights as a function of the relative timing of pre- and postsynaptic spikes (*source:* [6])

with different neural networks and neuron models. The second part is to estimate information flow of a neural network under STDP. Intuitively, synaptic plasticity and information flow are both closely related to the learning process, therefore the information flow through the network ought to evolve, in a measurable way, under synaptic plasticity. We choose "nearest neighbor" implementation of STDP among various implementations in [2].

3 Neuron models

3.1 Leaky Integrate-and-Fire model (LIF)

One of the simplest neuron model is leaky integrate-and-fire (LIF). The model describes the membrane potential v related to an input current $I(t)$ via the equation of local charge conservation:

$$\tau_m \frac{dv}{dt} = RI(t) + v_{rest} - v$$

equivalently, let $RI(t) = v_{in}(t)$, we have:

$$\tau_m \frac{dv}{dt} = v_{in} + v_{rest} - v \tag{1}$$

. In this model:

- τ_m: a time constant.
- $I(t)$: input current (from stimulus, or other neurons)
- R: electrical resistance of neuron
- u_{rest}: resting potential. If $I(t) \equiv 0$, v converges to v_{rest} with characteristic time τ_m.

$$u(t) = u_{rest} + e^{-t/\tau_m}(u(0) - u_{rest})$$

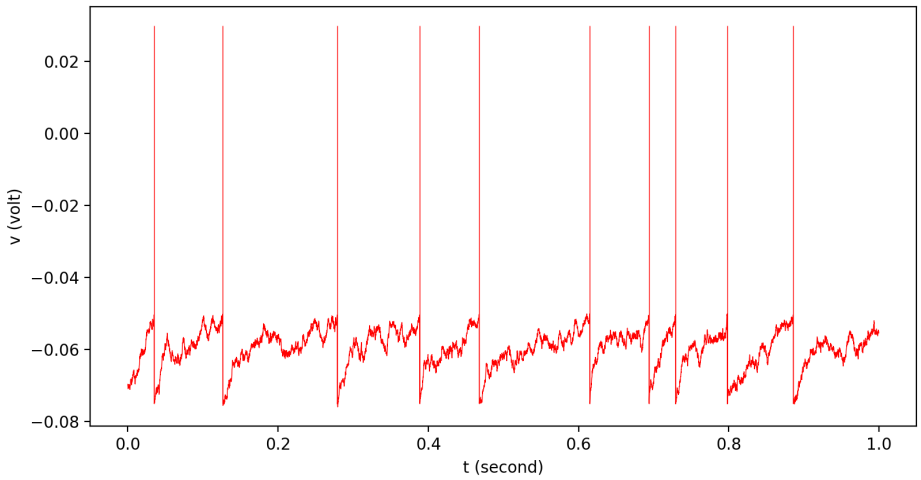

Figure 6: Simulation of LIF model using Brian simulator. $v_{rest} = -70mV, v_{thresh} = -50mV, v_{reset} = -75mV, \tau_m = 20ms, v_{in}(t) = \mu + \sigma\xi(t)$ where $\mu = 16mV, \sigma = 0.8mV.s^{-1/2}$ and $\xi(t)$ is standard white noise with (volt, second) as basic unit

.

Equation 1 is too simple that it cannot explain why a neuron spikes. To include the spiking phenomenon to our model we add the following rule: when the membrane potential reaches a threshold v_{thresh}, the neuron spikes and v is reset to a value v_{reset}. Figure 6 is a simulation of LIF model with white-noise input current.

Things that are simplified in LIF model ([13]):

- Spikes happen at a exact point of time. Spikes of a real neuron take place in a very short period of time (about $1ms$) compared to membrane time constant τ_m (about $20ms$).
- Firing threshold is a constant. It is not fixed for real neurons. For example, after a period of fast spiking, a neuron may become "tired" and is more difficult to spike again, which means the threshold becomes higher. Improved version of LIF model have a dynamic threshold that changes over time.
- The membrane potential resumes its dynamics immediately after a spike. In reality, after a spike, there is a refractory period in which neuron is insensitive to stimuli.

The LIF model is easy to implemented with low computational cost, but is incapable of producing rich spiking patterns of neurons.

3.2 Izhikevich model

(Figure 7). The Izhikevich model of neuron ([1]) can be written:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I$$
$$\frac{du}{dt} = a(bv - u)$$
$$\text{if } v \geq 30\text{mV, then } v \leftarrow c, u \leftarrow u + d \ (*)$$

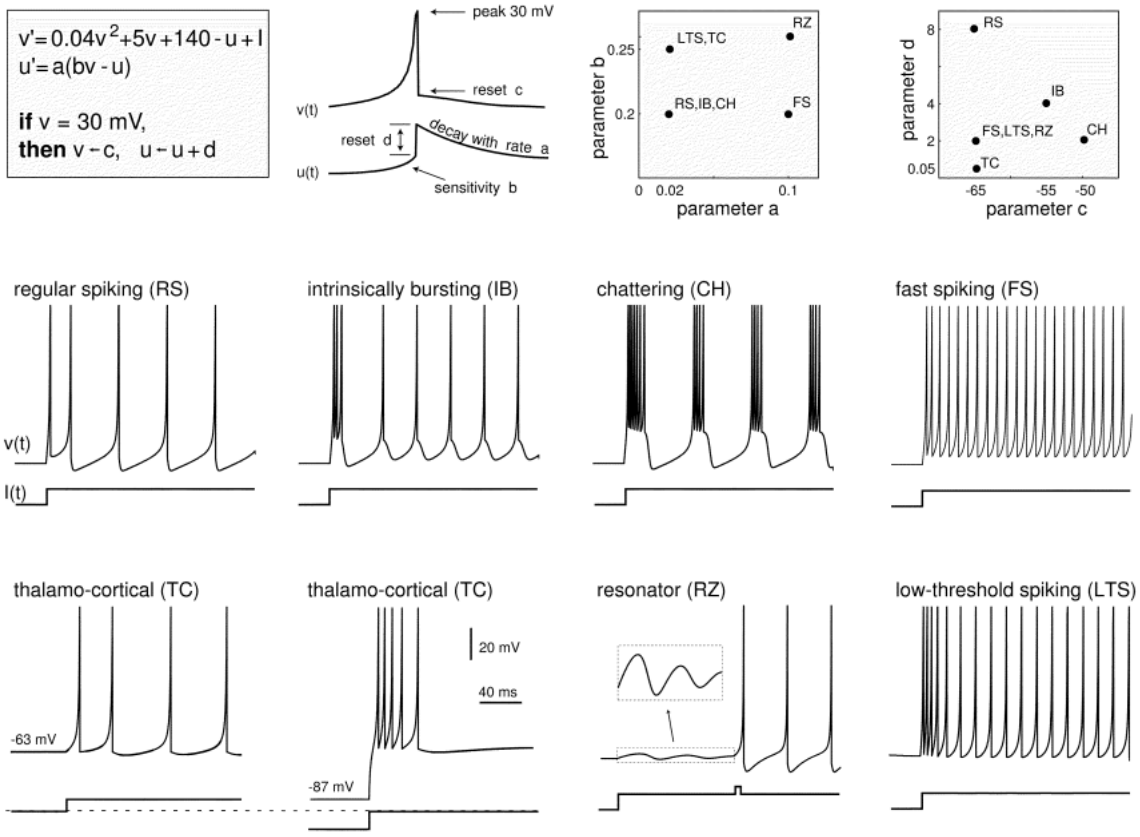

Figure 7: Firing patterns reproduced by Izhikevich model (*source:* [1])

where v represents the membrane potential and u accounts for the activation of K + ionic currents and inactivation of Na + ionic currents ([1]). After the spike reaches its peak (+ 30 mV), u and v are reset according to (*).

- The parameter a describes the time scale u.
- The parameter b describes the sensitivity of u to the fluctuation of v.
- The parameter c and d describes the after-spike reset of the membrane potential v and recovery variable u.

Izhikevich model is computationally simple and can reproduce all known firing patterns, depending on the parameters a, b, c, d (Figure 7)
The non-linearity of Izhikevich model makes it very different from LIF model. For example, if the threshold rule is removed from these models, then in response to a constant input, the membrane potential always converges to a finite value in LIF model while blows up for sufficiently strong current in Izhikevich model. Therefore the threshold serves two very different purposes in two models. In LIF model, the threshold is a critical voltage for spike initiation and has typical value of $-50mV$. In Izhikevich model, threshold is used to cut off the potential before resetting and has value of $30mV$.

4 Model of neuronal network

In this section we describe neuronal networks of n neurons in which n_e neuron are excitatory. The description consists of two parts: the connection is modeled by a graph and the interaction requires some further mathematical details about what happens when a neuron receives a spike.

4.1 Neuron connection

The connection between neurons is modeled by a directed graph $G(V, E)$ where $V = \{1, 2, ..., n\}$ and $E \subset \{(i, j)|i, j \in V\}$. The vertices V of graph represents neurons and the edge $(i, j) \in E$ represents the synapse $i \to j$. The edge weights $w_{i\to j} \in [0, \infty]$ represents the weight of synapse $i \to j$. Neurons $1, 2, ..., n_e$ are excitatory and the others are inhibitory.
We use Erdos-Renyi graph $G(n, p)$ ([17]) which is constructed by connecting nodes randomly, each edge is included in the graph with probability p independent from every other edge, with a realistic constraint that a neuron cannot connect to itself.

4.2 Interaction between neurons

We used the current-based model of neural network ([13], p.298). The input current of each neuron in the network consists of three sources:

- The current I_{ext} from all neurons outside the network.
- The current I_+ from all excitatory neurons inside the network.
- The current I_- from all inhibitory neurons inside the network.

More precisely, the equation of a neuron inside the network is given by:

$$\tau_m \frac{dv}{dt} = R(I_{ext} + I_+ + I_-) + v_{rest} - v$$

for LIF neuron, and

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I_{ext} + I_+ + I_-$$

for Izhikevich neuron.
The excitatory current entering neuron i is given by

$$I_+^i(t) = \sum_{\substack{j \text{ excitatory} \\ (j,i)\in E}} w_{j\to i} \sum_{s\in S^j, s<t} \alpha_+(t - s)$$

where S^j is the spike train of neuron j, and α_+ is called post-synaptic current. To simplify the calculation, we choose

$$\alpha_+(t) = 1_{t>0} c_+ \exp(-t/\tau_+)$$

where c_+, τ_+ represents the amplitude and decaying time constant of postsynaptic current. With this choice of α_+, I_+^i can be calculated efficiently as followings:

$$I_+^i(t) \leftarrow I_+^i(t) + c_+ \qquad \text{if } t \text{ is a spiking time of some excitatory presynaptic neuron } j \ (*)$$
$$I_+^i(t) = e^{-\delta/\tau_+} I_+^i(t - \delta) \qquad \text{otherwise } (**)$$

where δ is the time step of the simulation. (*) can be easily verified and (**) comes from the fact that the excitatory post-synaptic current has a decaying time constant τ_+.
Similarly for inhibitory current of neuron i

$$I_-^i(t) = \sum_{\substack{j \text{ inhibitory} \\ (j,i)\in E}} w_{j\to i} \sum_{s\in S^j, s<t} \alpha_-(t-s)$$
$$\alpha_-(t) = -1_{t>0} c_- \exp(-t/\tau_-)$$

for implementation

$$I_-^i(t) \leftarrow I_-^i(t) - w_{j\to i} c_- \qquad \text{if } t \text{ is a spiking time of some inhibitory presynaptic neuron } j$$
$$I_-^i(t) = e^{-\delta/\tau_-} I_-^i(t-\delta) \qquad \text{otherwise}$$

4.3 Evolution of synaptic weight via STDP

Consider the connection between presynaptic neuron i and postsynaptic neuron j with synaptic weight w. Consider t_i, t_j spiking times of i and j (among many spikes of i and j) and $t = t_j - t_i$
From the experimental result in Figure 5, the change in synaptic weight can be modeled by the equation ([2])

$$\Delta w/w = \begin{cases} A_+ e^{-t/\tau_+} & \text{if } t > 0 \\ A_- e^{-t/\tau_-} & \text{if } t < 0 \end{cases}$$

and w is updated ($w \leftarrow w + \Delta w$) at time t_j if $t_j > t_i$ and at time t_i if $t_i > t_j$.
Since the effect of STDP in subsection 1.2 is only measured for isolated pairs of spike of pre- and postsynaptic neuron, it is not clear how STDP should be applied to natural spike trains, which involve many spikes and many possible pairings of spikes. In the following sections, we present 2 ways to implement STDP on spike trains.

4.3.1 Classical implementation of STDP

Classical implementation takes into account the effect of all pairs of spikes ([2]). Synaptic weight is constant between 2 consecutive spikes and is only modified at spiking times. Let t_j a spiking time of postsynaptic neuron j, the weight change at time t_j is

$$\Delta w(t_j) = \sum_{t_i : t_i < t_j} \Delta w_{t_i,t_j} \tag{2}$$
$$= w(t_j) \sum_{t_i : t_i < t_j} A_+ e^{-(t_j - t_i)/\tau_+} \tag{3}$$

where w_{t_i,t_j} is the weight change corresponding to pair of spikes (t_i, t_j)
The sum $\sum_{t_i:t_i<t_j} A_+ e^{-(t_j-t_i)/\tau_+}$ can be calculated efficiently as following. Let $x_+(t)$ such that $x_+(0) = 0$ and

$$\dot{x}_+(t) = -x_+(t)/\tau_+$$

if t is not a spiking time of i, and:

$$x_+(t) \leftarrow x_+(t) + A_+$$

if t is a spiking time of i. We can verify that $\sum_{t_i:t_i<t} A_+ e^{-(t-t_i)/\tau_+}$ and $x_+(t)$ satisfy the same initial condition and differential equation, therefore are identical. It is easier to calculate $x_+(t)$ since we do not need to sum over all $t_i < t$ for each t. The weight change at time t_j can be rewritten as

$$\Delta w(t_j) = w(t_j) x_+(t_j)$$

Similarly, let t_i a spiking time of i. The weight change at time t_i is:

$$\begin{aligned}\Delta w(t_i) &= \sum_{t_j:t_j<t_i} \Delta w_{t_i,t_j} \\ &= w(t_i) \sum_{t_j:t_j<t_i} A_- e^{-(t_i-t_j)/\tau_-}\end{aligned}$$

Let $x_-(t)$ such that $x_-(0) = 0$ and

$$\dot{x}_-(t) = -x_-(t)/\tau_-$$

if t is not a spiking time of j, and:

$$x_-(t) \leftarrow x_-(t) + A_-$$

if t is a spiking time of j. The we have

$$\Delta w_{t_i} = w(t_i) x_-(t_i)$$

4.3.2 Nearest-neighbor STDP

Since the membrane potential is reset after each spike, the most recent spike overrides the effect of all previous spikes (of the same neuron). Therefore it is more reasonable to consider only the nearest-neighbor pairs. To calculate the weight change at t_j, we only consider the nearest presynaptic spike before t_j ([2]).

$$\Delta w_{t_j} = w_{t_i^*,t_j} = w(t_j) A_+ e^{-(t_j-t_i^*)/\tau_+}$$

where $t_i^* = max\{t_i : t_i < t_j\}$
Similarly:

$$\Delta w_{t_i} = w_{t_j^*,t_i} = w(t_i) A_- e^{-(t_i-t_j^*)/\tau_-}$$

where $t_j^* = max\{t_j : t_j < t_i\}$
The implementation is mostly the same as the classical case, except for the update rule of x_+ and x_- at spiking times:

$$x_+(t) \leftarrow A_+$$

if t is a spiking time of i, and

$$x_-(t) \leftarrow A_-$$

if t is a spiking time of j.

5 Probabilistic model of spiking activity

In this section, we try to model the spiking activity by a point process named multivariate Hawkes process. A point process is a collection of random points on some underlying mathematical space such as the real line, the Cartesian plane, or more abstract spaces. For example, to model the spiking time of a neuron, the underlying space is a time interval.

5.1 Poisson process

To understand Hawkes process, we need to understand Poisson process first.
A Poisson process is a collection of random points in $\mathbb{R}_+$. For example, it can be use to model a series of event that randomly happens over time. Let $N(t)$ the number of points in interval $[0, t]$. A Poisson process with **intensity function** (**rate function**) $\lambda(t)$ satisfies, for all $h = o(1)$:

$$N(t+h) - N(t) = \begin{cases} 1, & p = \lambda(t)h + o(h) \\ 0, & p = 1 - \lambda(t)h + o(h) \end{cases}$$

Moreover, if $[t, t+h]$ and $[s, s+h]$ are 2 separated intervals then $N(t+h)-N(t)$ and $N(s+h)-N(s)$ are independent. The intensity function measures how likely a point appears in a specific location: the probability of finding a point in a small interval $[t, t+h]$ is $\lambda(t)h + o(h)$.

5.2 Self-exciting process

Self-exciting processes take into account the causal relation between events. Each event can trigger another event. From another point of view, the likelihood of an event happening at time t depends on all the events that happened before t.
The name "self-exciting" comes from the co-dependence between the point process and its intensity function. We know that the intensity function "generates" the point process. Contrary to the Poisson process where the intensity function is fixed, intensity function of self-exciting process is itself "generated" by the point process:

$$\begin{aligned} \lambda(t) &= \nu(t) + \int_{-\infty}^{t} h(t-s) dN_s \\ &= \nu(t) + \sum_{t_k < t} h(t - t_k) \end{aligned}$$

where $(t_k)_{k \in \mathbb{N}}$ is the point process, $\nu(t)$ represents a background intensity and function $h : \mathbb{R} \to \mathbb{R}^+$ represents the impact of each event. Causality imposes that $h(t) = 0$ for $t < 0$. In the case of self-exciting process, h only takes non-negative value. If an event can inhibit other events in future then h can take negative value. To keep the intensity function positive, we add a modification to the previous definition:

$$\lambda(t) = \big(\nu(t) + \sum_{t_k < t} h(t - t_k)\big)_+$$

where $x_+ = 0$ if $x < 0$ and $x_+ = x$ if $x \geq 0$.

5.3 Multivariate Hawkes process

Consider a neural network (several neurons measured in an experiment) embedded in a larger neural network (e.g. the whole brain). All we know about this sub network is its spiking activity. To infer some useful information about this network, we need a probabilistic model that relates the inner structure of the network and its spiking activity. A suitable model is multivariate Hawkes process. In this model, each neuron i has a background spiking intensity ν_i, caused by neurons outside the network. We know that when a neuron spikes, it exert an impact on its own spiking activity and on spiking activity of its output neurons. The impact of a neuron j on neuron i is modeled by a real

function $h_{j\to i}(t)$. Similarly to the self-exciting point process, the intensity function $\lambda_i(t)$ of neuron i is given by:

$$\lambda_i(t) = \big(\nu_i + \sum_{j=1}^{N} \sum_{T\in N^j, T<t} h_{j\to i(t-T)}\big)_+$$

where N_j is the spike train of neuron j.
Since interactions between neurons can be excitatory or inhibitory, $h_{j\to i}$ can be positive or negative. Causality imposes that $h_{j\to i}$ vanishes on $(-\infty, 0]$.
Now we assume that the impact of inhibitory spikes are sufficiently weak so that the value inside the $(.)_+$ function of intensity function never reaches zero. Therefore

$$\lambda_i(t) = \nu_i + \sum_{j=1}^{N} \sum_{T\in N^j, T<t} h_{j\to i(t-T)}$$

Now the model is linear and therefore more amenable to mathematical analysis.

6 Reconstructing functional connectivity of neural network

Imagine a neural network whose spiking activity is a true Hawkes process with parameters $(\boldsymbol{\nu_0}, \boldsymbol{h_0})$. Suppose we observed its spikes in $[0, T]$. Let N_t^i be the number of spikes of neuron i in $[0, t]$. We want to estimate ν_j and $h_{j\to i}$ for $i, j = 1, 2, ...n$ where n is the number of neurons.

6.1 Least square method

Let us start with a more simple question: consider a Poisson process with intensity function $\lambda_0(t)$ and its data observed in $[0, T]$ with N_t is the number of points in $[0, t]$. How to estimate λ_0?
Let $\lambda \in L^2([0, T])$. Consider the random variable:

$$J(\lambda) = -2\int_0^T \lambda(t)dN_t + \int_0^T \lambda^2(t)dt$$

then we have:

$$\lambda_0 = \text{argmin}_\lambda \mathbb{E}\big[J(\lambda)\big] \tag{4}$$

in fact, using $\mathbb{E}[dN_t] = \lambda_0(t)dt$ we have

$$\begin{aligned}
\mathbb{E}\big[J(\lambda)\big] &= \mathbb{E}\big[-2\int_0^T \lambda(t)dN_t + \int_0^T \lambda^2(t)dt\big] \\
&= -2\int_0^T \lambda(t)\mathbb{E}[dN_t] + \int_0^T \lambda^2(t)dt \\
&= -2\int_0^T \lambda(t)\lambda_0(t)dt + \int_0^T \lambda^2(t)dt \\
&= ||\lambda - \lambda_0||_2^2 - ||\lambda_0||_2^2
\end{aligned}$$

from which follows Equation 4. A "reasonable" estimator for λ_0 is

$$\hat{\lambda} = \text{argmin}\mathbb{E}\big[J(\lambda)\big]$$

Here we $\mathbb{E}$ is the empirical mean.
In general, let us consider a statistical model with parameter θ and some empirical criterion J such that $\theta \to \mathbb{E}_{\theta_0}[J(\theta)]$ achieves a minimum at point θ_0. Such a criterion is called an **empirical contrast** for the estimation of θ_0.
Now let us go back to the original statistical problem for multivariate Hawkes model.
Let $\boldsymbol{\nu} = (\nu_i)_{i=1,..,n}$, $\boldsymbol{h} = (h_{j\to i})_{i,j=1,...,n}$.
Remind that N_t^i is a random process associated with parameters $(\boldsymbol{\nu_0}, \boldsymbol{h_0})$, not $(\boldsymbol{\nu}, \boldsymbol{h})$
Similarly to the case of Poisson process the previous example, we can use a minimum contrast estimator to estimate $(\boldsymbol{\nu_0}, \boldsymbol{h_0})$. However, we observe only one realization of the Hawkes process in a time interval $[0, T]$. Therefore we use the following estimator

$$(\hat{\nu_i}, (\hat{h}_{j\to i})_{j=1,...,n}) = \text{argmin} J_i(\nu_i, (h_{j\to i})_{j=1,...,n}) \tag{5}$$

where

$$J_i(\nu_i, (h_{j\to i})_{j=1,...,n}) = -2\int_0^T \lambda_i(t) dN_t^i + \int_0^T \lambda_i^2(t) dt$$

and

$$\lambda_i(t) = \nu_i + \sum_{j=1}^{N} \sum_{T\in N^j, T<t} h_{j\to i}(t-T)$$

By minimizing J_i for $i = 1, ..., n$, we obtain the estimations $(\boldsymbol{\nu}, \boldsymbol{h})$ for the parameters $(\boldsymbol{\nu_0}, \boldsymbol{h_0})$ of Hawkes process.

We solve Equation 5 numerically in the space of functions $h_{j\to i}$ that are constant in each interval $[k\delta, (k+1)\delta]$, for $k = 0, ..., K-1$ and vanish outside the interval $[0, K\delta]$:

$$h_{j\to i}(t) = \sum_{k=1}^{K} a_{j\to i}^k 1_{((k-1)\delta,\ k\delta]}(t)$$

Let

$$\boldsymbol{a}_i = \left(\nu_i, (a_{j\to i}^k)_{k=1,...,K\ j=1,...,N}\right)$$

Then we have

$$\begin{aligned}\lambda_i(t) &= \nu_i + \sum_{j=1}^{N} \sum_{T\in N^j,\ T<t} h_{j\to i}(t-T) \\ &= \nu_i + \sum_{j=1}^{N} \sum_{k=1}^{K} a_{j\to i}^k N^j_{[t-k\delta,\ t-(k-1)\delta)}\end{aligned}$$

Let

$$\boldsymbol{c}_t = \left(1, (N^j_{[t-k\delta,\ t-(k-1)\delta)})_{k=1,...,K\ j=1,...,N}\right)$$

then

$$\lambda_i(t) = \boldsymbol{c}_t^T \boldsymbol{a}_i$$

The problem of minimizing $\int_0^T \lambda_i^2(t)dt - 2\int_0^T \lambda_i(t) dN_t^i$ is transformed into minimizing

$$\int_0^T \boldsymbol{a}_i^T \boldsymbol{c}_t \boldsymbol{c}_t^T \boldsymbol{a}_i dt - 2\int_0^T \boldsymbol{a}_i^T \boldsymbol{c}_t dN_t^i$$

or

$$\boldsymbol{a}_i^T \boldsymbol{G} \boldsymbol{a}_i - 2\boldsymbol{a}_{\boldsymbol{i}}^T \boldsymbol{b}_i$$

where:

$$\boldsymbol{G} = \int_0^T \boldsymbol{c}_t \boldsymbol{c}_t^T dt$$
$$\boldsymbol{b}_i = \int_0^T \boldsymbol{c}_t^T dN_t^i$$

G is invertible with high probability ([4]). Therefore the least square estimate of the parameter $\boldsymbol{a}_i$ is given by

$$\hat{\boldsymbol{a}}_i = \boldsymbol{G}^{-1} \boldsymbol{b}_i$$

6.2 LASSO estimation

To gain sparsity, we use the following estimator:

$$\hat{\boldsymbol{a}}_i = \text{argmin}_\beta - 2\beta^T \boldsymbol{b}_i + \beta^T \boldsymbol{G} \beta + 2\boldsymbol{d}_i^T |\beta|$$

where $|\beta| = (|\beta_i|)_i$ and $\boldsymbol{d}_i$ is a vector of weights. In [3] the following data-dependent weights are used based on an estimation of the variance on the data themselves:

$$\boldsymbol{d}_i = \sqrt{2\gamma \log(n + n^2 K) \int_0^T \boldsymbol{c}_t^2 d\boldsymbol{N}_i(t)} + \frac{\gamma \log(n + n^2 K)}{3} \sup_{t \in [0,T]} |\boldsymbol{c}_t|$$

The theory behind this choice of weights is fully detailed in [4].
Note that $\gamma = 0$ corresponds to the least-square method where all estimated coefficients are nonzero and that the sparsity (the proportion of zero coefficients) increases with γ.
We define the the weights of reconstructed connections as L^1-norm of corresponding interaction functions.

7 Information theory

7.1 Entropy

Definition The entropy $H(X)$ of a discrete random variable X is define by

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$$

where $\mathcal{X}$ is the set of values of X. The log is to the base 2 and entropy is expressed in bits. For example, the entropy of a fair coin toss is 1 bit. We use the convention $0 \log 0 = 0$, which is justified by the fact that $x \log x \to 0$ as $x \to 0$.

In information theory, the analog of the law of large numbers is the **asymptotic equipartition property**
Theorem ([7]) If $X_1, X_2, ...$ are i.i.d. $\sim p(x)$, then

$$-\frac{1}{n} \log p(X_1, X_2, ..., X_n) \to H(X)$$

in probability. Equivalently, we have the following theorem
Theorem ([7]) Given any $\epsilon > 0$ and $\delta > 0$, we can find n_0 sufficiently large so that the sequences of length $n \geq n_0$ falls into two classes:

1. A set whose total probability is less than ϵ.

2. The remainder, all of whose members have probabilities satisfying:

$$\left| \frac{\log p^{-1}}{N} - H \right| < \delta$$

This theorem says that, for n sufficiently large, almost all sequences $(X_1, X_2, ..., X_n)$ are almost equally surprising. It is possible for most purposes to treat the long sequences of length n as though there were just 2^{nH} of them, each with a probability 2^{-nH}. This means, from the point of view of information theory, a sequence of n i.i.d. X has the same information content as a sequence of nH random coin toss, or nH bits, therefore the random variable X has the information content of H bits.

7.2 Joint entropy, conditional entropy and mutual information

Definition The **joint entropy** $H(X, Y)$ of a pair of discrete random variable (X, Y) with a joint distribution $p(x, y)$ is defined as

$$H(X, Y) = -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log p(x, y)$$

Definition The conditional entropy $H(Y|X)$ is defined as

$$H(Y|X) = \sum_{x \in \mathcal{X}} p(x) H(Y|X = x)$$

where

$$H(Y|X = x) = -\sum_{y \in \mathcal{Y}} p(y|x) \log p(y|x)$$

Theorem (*Chain rule*)

$$H(X, Y) = H(X) + H(Y|X)$$

Definition The **mutual information** $I(X, Y)$ of a pair of discrete random variable (X, Y) is defined as

$$\begin{aligned} I(X, Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X, Y) \end{aligned}$$

equivalently

$$I(X, Y) = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}$$

The relations between entropy, conditional entropy and mutual information is summarized by Figure 8

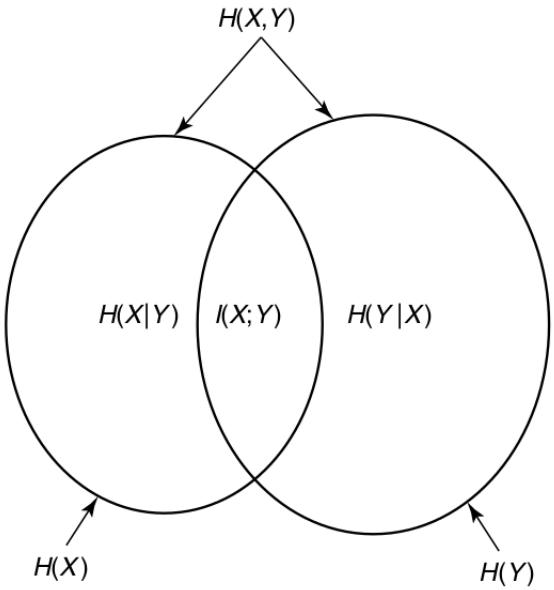

Figure 8: Relationship between entropy and mutual information: $H(X) = H(X|Y) + I(X,Y)$, $H(Y) = H(Y|X) + I(X,Y)$,etc

8 Estimating information flow in neural network

8.1 First method: direct estimation from spike train

Because a spike train can be arbitrarily long, by mutual information (entropy) we mean mutual information (entropy) per time unit, measured in bits per second. Consider a spike train X. By dividing time into small intervals of size δ such that there is at most one spike in each interval, X can be transformed into a sequence $(X_i)_{i\in\mathbb{N}}$ where $X_i = 1$ if there is a spike in interval i and 0 otherwise. Suppose X is stationary so that X_i are identically distributed and $X_i \sim X^* \ \forall i$. For convenience, we use X to denote both the spike train and the sequence (X_i)

Suppose we want to estimate the mutual information between 2 spike trains X, Y. We might define the mutual information as $I(X^*, Y^*)/\delta$. However, this definition of mutual information is not satisfying. Let ϕ be a permutation of $\{0,1\}^{10}$ and $\phi(X)$ the sequence obtained by applying ϕ to each block of size 10 of X. Since there is a unique way to transform X to Y and Y to X, a good definition of mutual information must satisfy $H(X) = H(Y) = I(X,Y)$. However, if we know one bit of X_i, in general there is no way to know for sure the corresponding bit Y_i, therefore the estimation by formula $I(X^*, Y^*)/\delta$ will result in $I(X,Y) < I(X)$ which is inaccurate.

We can estimate $I(X,Y)$ by considering them as sequences $(X_i), (Y_i)$ of k-bit words so that the time window $k\delta$ is large enough, for example $k = 20$ and $\delta = 3ms$, and apply the same method as above. In a naive way, to estimate $\mathbb{P}(X_i = x, Y_i = y)$ for all $x, y \in \{0,1\}^k$, we can accumulate $\sim 2^{2k}$ samples of (X_i, Y_i) which requires a simulation time of $\sim \delta k 2^{2k}$. This is a very large number even for $k = 20$ and $\delta = 3ms$.
In [9], [19] a method is proposed to extrapolate the value of the mutual information when $k\delta \to \infty$ However, due to lack of time, we have not implemented this method.

8.2 Second method: estimating information flow via Hawkes model

8.2.1 Estimating correlations between neurons via Hawkes model

Consider a neural network whose spiking activity is a true Hawkes process with background intensity $\nu = (\nu_i)_{i=1,\dots,n}$, and interaction functions $\boldsymbol{h} = (h_{ij})_{i,j=1,\dots,n} = (h_{j\to i})_{i,j=1,\dots,n}$.
Let S_i be the set of spiking times of neuron i and $N_i(t)$ be the number of spikes of neuron i in $[0,t]$, then

$$\frac{dN_i(t)}{dt} = \sum_{s\in S_i} \delta(t-s)$$

where δ is Dirac delta-function.
Denote $(f*g)(t) = \int_{-\infty}^{\infty} f(s)g(t-s)ds$. If $g(t) = \sum_{s\in S}\delta(t-s)$ for some set S then

$$(f*g)(t) = \sum_{s\in S} f(t-s)$$

The intensity function of neuron i is

$$\begin{aligned}\lambda_i(t) &= \nu_i + \sum_j \sum_{s\in S_j} h_{j\to i}(t-s)\\ &= \nu_i + \sum_j (h_{j\to i} * dN_j)(t)\\ &= (\boldsymbol{h} * d\boldsymbol{N})_i(t)\end{aligned}$$

where $\boldsymbol{N} = (N_i)_{i=1,\dots,n}$. Equivalently:

$$\boldsymbol{\lambda}(t) = (\boldsymbol{h} * d\boldsymbol{N})(t)$$

In the equilibrium state, the expected value of intensity function $\lambda_i(t)$ does not depend on time, let

$$\bar{\lambda}_i = \langle \lambda_i(t)\rangle$$

and $\bar{\boldsymbol{\lambda}} = (\bar{\lambda}_i)_{i=1,\dots,n}$. Taking the expectation on two sides of the identity

$$\lambda_i(t) = \nu_i + \sum_j \int_{-\infty}^{\infty} h_{j\to i}(s) dN_j(t-s)$$

we get

$$\bar{\lambda}_i = \nu_i + \sum_j \int_{-\infty}^{\infty} h_{j\to i}(s)\bar{\lambda}_j$$

or

$$\bar{\boldsymbol{\lambda}} = \boldsymbol{\nu} + \big(\int_{-\infty}^{\infty} \boldsymbol{h}(s)ds\big)\bar{\boldsymbol{\lambda}}$$

therefore

$$\bar{\boldsymbol{\lambda}} = \big(\mathbf{1} - \int_{-\infty}^{\infty} \boldsymbol{h}(s)ds\big)^{-1}\boldsymbol{\nu}$$

where $\mathbf{1}$ is the identity matrix.
The correlation between spike train i and j with delay time τ is defined as

$$\begin{aligned} C_{ij}(\tau) &= \langle \frac{dN_i(t+\tau)}{dt} \frac{dN_j(t)}{dt} \rangle - \langle \frac{dN_i(t+\tau)}{dt} \rangle \langle \frac{dN_j(t)}{dt} \rangle \\ &= \langle \frac{dN_i(t+\tau)}{dt} \frac{dN_j(t)}{dt} \rangle - \bar{\lambda}_i \bar{\lambda}_j \end{aligned}$$

and $\boldsymbol{C}(\tau) = (C_{ij}(\tau))_{i,j=1,\dots,n}$ is called **covariance density matrix**.
A key result in [12] is that, given the Fourier transform of $\boldsymbol{h}$:

$$\hat{\boldsymbol{h}}(\omega) = \int_{-\infty}^{\infty} e^{-i\omega t} \boldsymbol{h}(t) dt$$

the Fourier transform of the covariance density matrix $\boldsymbol{C}$ is given by

$$\hat{\boldsymbol{C}}(\omega) = \big[\mathbf{1} - \hat{\boldsymbol{h}}(\omega)\big]^{-1} \bar{\boldsymbol{\lambda}} \big[\mathbf{1} - \hat{\boldsymbol{h}}^T(-\omega)\big]^{-1} \tag{6}$$

8.2.2 Calculating mutual information via correlations

We still do not know whether mutual information can be calculated via correlation densities.

9 Results

9.1 Performance of LASSO method

We test the performance of LASSO method ([4], [3]) with LIF and Izhikevich neural networks, each has 15 neurons. For each $p \in \{0.05, 0.1, 0.15\}$ and $n_e \in \{0, 5, 10\}$, we simulate 100 neural networks, each simulation has 1000 spikes per neuron on average. All LIF neurons have the same parameters $v_{rest} = -70mV, v_{thresh} = -50mV, v_{reset} = -75mV, \tau_m = 20ms$ (to emulate the shape of action potential in Figure 2, v_reset is usually chosen to be lower than v_rest). Each neuron i receives an input current I^i such that $RI^i(t) = \mu + \sigma\xi^i(t)$ where $\xi^i(t)$ is standard white noise [1] and ξ^i is independent with ξ^j if $i \neq j$. Parameters μ and σ are chosen so that the firing frequency of a single neuron receiving the noise input is about $10 - 15Hz$: we take $\mu = 16mV, \sigma = 0.8mV.s^{-1/2}$ [2] . The parameters c_+, τ_+ are chosen in such a way that four presynaptic spikes at frequency $100Hz$ are enough to generate one postsynaptic spike. Here we take $Rc_+ = Rc_- = 50mV, \tau_+ = \tau_- = 5ms$.

All Izhikevich neurons are regular spiking neuron with parameters $a = 0.02, b = 0.2, c = -65, d = 3$. The parameters μ, σ, c_+, τ_+ are chosen using the same criteria as for LIF model, except that three presynaptic spikes at frequency $100Hz$ are enough to generate one post synaptic spike [3] . Here we take $\mu = 3.5V.s^{-1}, \sigma = 0.1V.s^{-1/2}$ [4], $c_+ = c_- = 5.5V.s^{-1}, \tau_+ = \tau_- = 5ms$.

[1] $\xi(t) = dB(t)/dt$ where $B(t)$ is Brownian motion and t is measured in seconds. Since $\langle B(t)^2 \rangle = t$, $B(t)$ has unit $s^{1/2}$ and $\xi(t)$ has unit $s^{-1/2}$

[2] Since ξ has unit $s^{1/2}$, σ has unit $V.s^{-1/2}$ so that $RI = \mu + \sigma\xi$ has unit V

[3] This is the main reason why the performance of LASSO for Izhikevich model is better 1. Due to lack of time we cannot rerun the simulations of LIF networks with different parameters so that this property is also true for LIF model

[4] The unit and order of magnitude of σ and μ in Izhikevich model are different from LIF model because $I(t)$ in Izhikevich model has unit $V.s^{-1}$, not A (Ampere)

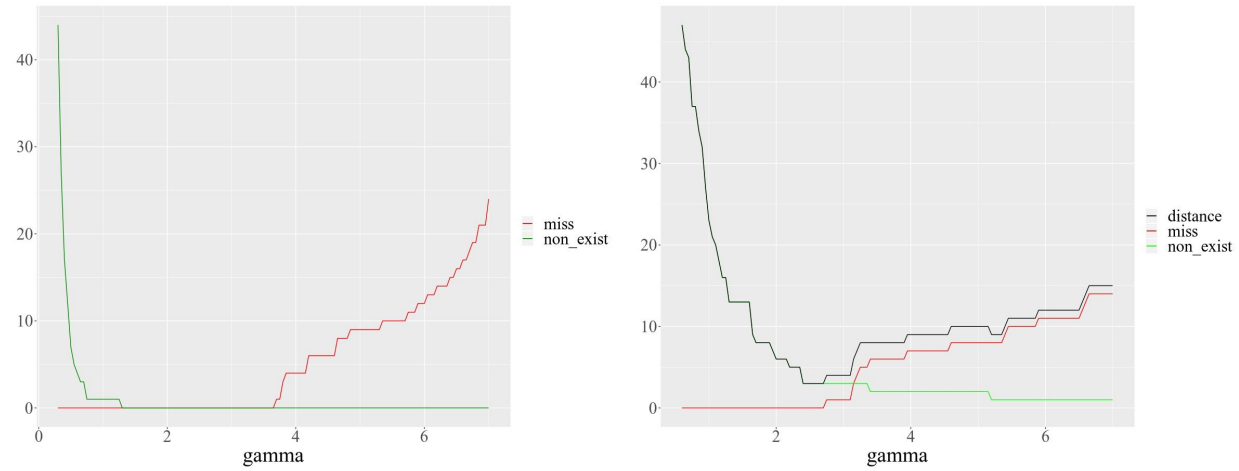

Figure 9: Dependence of number of missing and non-existing edges on γ when perfect reconstruction is possible (left) and not possible (right). Here $\delta = 2ms$ and LASSO is applied to 2 networks with different connectivity.

The LASSO method has three parameters K, δ and γ. We fix $K = 5$ and for each simulation we find δ, γ that minimize the distance between the reconstructed and the original connection. Here the distance between the original $G = (V, E)$ and the reconstructed $G' = (V, E')$ is defined as $d(G, G') = |(E \setminus E') \cup (E' \setminus E)|$. We call $E \setminus E'$ the set of "missing" edges and $E' \setminus E$ "non-existing" edges. Note that when considering E and E' we ignored all self connections, since neuron cannot connect to itself. The reconstructed graph always has negative self-connections, which we will explain later.
The value of δ is chosen from $\{2, 3, 4, 5, 6, 7\}(ms)$. For a fixed value of δ, as γ increases, the number of non-existing edges decreases and the number of missing edges increases, and the minimum distance is achieved near the intersection of two curves (Figure 9). Since the distance has integer values and γ has positive real values, we can find intervals of γ on which the minimum distance is achieved. Among all of these intervals, let $[\gamma_a, \gamma_b]$ be the first interval to the left.

For each spiking data of a single simulation, by connecting the values of γ_a, γ_b corresponding to each δ, we can approximately find region of (δ, γ) where the best reconstruction is attained (Figure 10)

For 100 data-sets corresponding to fixed values of n_e and p, we superimpose all the optimal regions to find the region in which the best performance is achieved for all graphs of the same class. (Figure 11). For graphs with $(n_e, p) = (0, 0.05)$, a common perfect reconstruction can be achieved by taking $\gamma = 2.5$. For other cases, for example $(n_e, \delta) = (10, 0.15)$, one best choice of (δ, γ) is not achievable. In general, $\delta = 2ms$ works better than other values because at this value of δ, the red and blue points are well separated. For all data-sets corresponding to each value of (n_e, p), we calculate the mean and standard deviation of distances corresponding to the best performance. The result is given in the Table 1
There are many factors that are fixed or ignored in our simulations, but might have an important effect on the performance of LASSO.

- The amplitude of postsynaptic current, characterized by the constants c_+, c_-. If the postsynaptic current is very weak, it is difficult to detect the connections between neurons.
- The time constant τ_+, τ_- of post synaptic current, which affects the choice of K and *delta* in

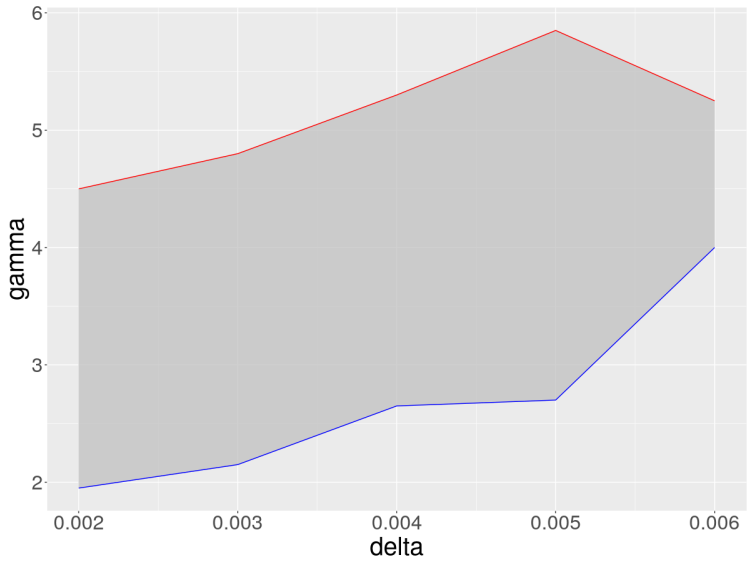

Figure 10: Example of a region of (δ, γ) where the minimum distance is attained. The region is calculated for a single simulation of Izhikevich network with $n_e = 5, p = 0.1$. The minimum distance is 0

.

Table 1: Mean and standard deviation of distance for each (n_e, p), Izhikevich model(left) and LIF (right) model

$n_e; p$	0.05	0.10	0.15
0	(0, 0)	(0,0	(0.0)
5	(0.01, 0.1)	(0.1, 0.48)	(0.86, 1.55)
10	(0, 0)	(0.49, 1.01)	(4.02, 4.36))

$n_e; p$	0.05	0.10	0.15
0	(0, 0)	(0,0	(0.0)
5	(0.04, 0.19)	(0.63, 1.81)	(3.06, 0.29)
10	(0.07, 0.29)	(2.43, 3.75)	(7.9, 6.48))

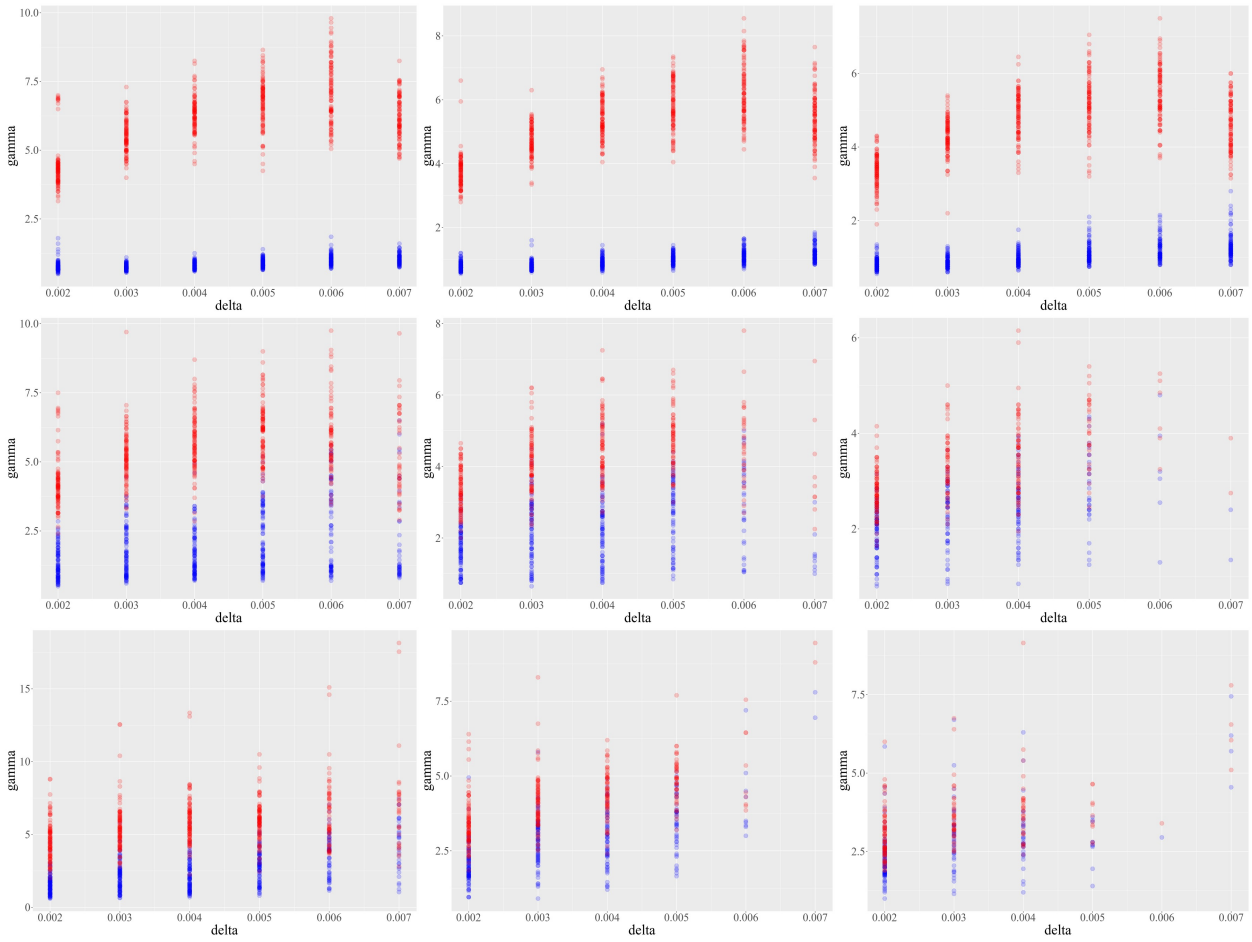

Figure 11: Superimposition of all optimal regions of all graph of the same class. From left to right, up to down: (n_e, p) = (0, 0.05), (0, 0.10), (0, 0.15), (5, 0.05), (5, 0.10), (5, 0.15),(10, 0.05), (10, 0.10), (10, 0.15). Blue point, red point corresponds to γ_a, γ_b respectively

LASSO method.

- The axonal conductance delay, whose value ranges from $\sim 0.1ms$ to $\sim 10ms$ ([16]). We do not include this in the simulations. Applying LASSO to models with conductance delay will requires larger values of $K\delta$, which results in more computation for a fixed value of δ.

- The input current. In Hawkes model the spontaneous spiking is modeled by Poisson process with constant rate. However, when a neuron is injected with a white noise current, its spiking activity is statistically different from a Poisson process. Figure 12 compares the distribution of inter-spike time intervals of a LIF neuron receiving a white noise current to that of a Poisson process with the same average frequency. In contrast to Poisson process, the inter-spike intervals of the neuron is bounded away from zero. This explains the negative self-connections of the reconstructed graphs. These negative self-connections may have even bigger weights than the actual inhibitory connections. Since the LASSO method has tendency to filter out the connections that are weak compared to the others to achieve sparsity, this can make the detection of the inhibitory connections difficult. However, if we use a kind of input current that can generate Poissonian-like spike trains, the reconstruction will be better. For example, if a neuron is stimulated by an excitatory Poissonian spike train, its firing activity will be statistically close to a Poisson process.

- The difference between excitatory and inhibitory neurons. In our models, excitatory and inhibitory neurons have the same parameters. In more realistic simulations, inhibitory neurons are of fast-spiking type and excitatory neurons regular spiking type ([16]).

- The norm used to measure the distance between reconstructed and original graph. The counting distance that we use sometimes does not reflect the true quality of the reconstruction. For example, a reconstruction using ordinary least square method (OLS) ($\gamma = 0$) has all non-zero weights, so its performance is far worse than LASSO method in terms of counting distance, but in L^2 for example, their performances can be nearly equivalent. Figure 13 shows that LASSO is only significantly better when all the connections are inhibitory. With higher proportion of excitatory neuron, their performance are nearly equivalent in L^2 norm.

9.2 STDP simulation

The network structure in this section follows closely the STDP networks simulated in [16], [18]. A 20-neuron network is simulated for 30 seconds with 1 ms time step . There are 16 Regular Spiking excitatory neurons with $a = 0.02, b = 0.2, c = -65, d = 8$ and 4 Fast Spiking inhibitory neurons with parameters $a = 0.1, b = 0.2, c = -65, d = 2$. Each excitatory neuron connects randomly to 4 excitatory neurons and 1 inhibitory neuron. Each inhibitory neuron connects randomly to 5 excitatory neurons. There is no connection between inhibitory neurons. Initial weights are equal to 1 for all connections. Synaptic delays are chosen randomly from 1 ms to 15 ms. Parameters $\tau_+ = \tau_- = 32ms, A_+ = 0.01, A_- = -0.01$ are used for STDP. Inhibitory connections are not plastic. The weights of excitatory neurons are clipped to $[0, 2]$. Each neuron receives an constant input current $I(t) = 3.5V.s^{-1}$. The parameters of post-synaptic current are $c_+ = c_- = 5.5V.s^{-1}, \tau_+ = \tau_- = 5ms$. The neurons are expected to spike in asynchronous, time-locked patterns ([16]). Although we did not quantitatively determine the time-lock patterns, one can see them in Figure 14. InFigure 15 we plot the evolution of synaptic weights.

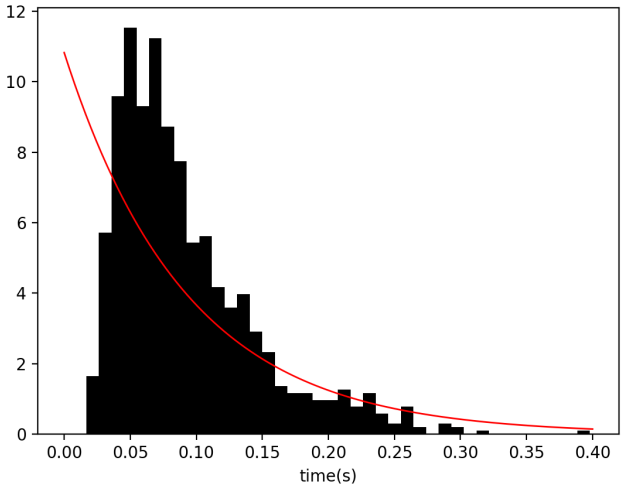

Figure 12: Inter-spike statistics of Poisson process and spiking activity of LIF neuron injected by noise current

9.3 Correlations in Hawkes network

We calculate the correlation densities of simple 3-neuron Hawkes networks using Equation 6. In Figure 16, Figure 17, Figure 18, Figure 19, different combinations of connectivity and synaptic weights are explored, without concern for the biological plausibility of the network. The nonzero interaction functions are exponential $h_{j\to i}(t) = Ce^{-t/t_0}/t_0$ where $C = \pm 0.5$, $t_0 = 5ms$ and background intensity is $10Hz$ for all neurons. The amplitude C characterize the average number of spikes generated by one spike.
By definition of $C_{j\to i}$ one have $C_{j\to i}(\tau) = C_{i\to j}(-\tau)$ and $C_{i\to i}(\tau) \to \infty$ as $\tau \to 0$. We can see that if the interaction functions $h_{i\to j}$ and $h_{j\to k}$ are nonzero and have the same amplitude, then $h_{i\to k}$ is nonzero with a smaller amplitude. Similarly, if $h_{i\to j}$ and $h_{i\to k}$ are nonzero with the same amplitude, then $h_{j\to k}$ and $h_{k\to j}$ are nonzero with a smaller amplitude.

10 Discussion

To estimate how information flow evolves in a plastic neural network, we need to divide time into intervals that are (1) small enough so that the synaptic weights do not change too much during a time interval and (2) large enough so that we can collect enough data to estimate the mutual information. LASSO method need $\sim 10^3$ spikes per neuron to achieve a good reconstruction of interaction functions, which means if the average firing rate is $\sim 10Hz$, we need to choose the STDP parameters A_+, A_- so that the relative changes of synaptic weights are small enough (about 10 % on average) during $\sim 10^2$ seconds. Measuring information flow when synaptic weights change quite rapidly is very challenging since we have to use smaller time intervals so that (1) is satisfied but then we do not have enough data in each interval to estimate the mutual information. We might increase the firing rates of neurons to increase the number of spikes in each interval but this will make synaptic weights change more rapidly.

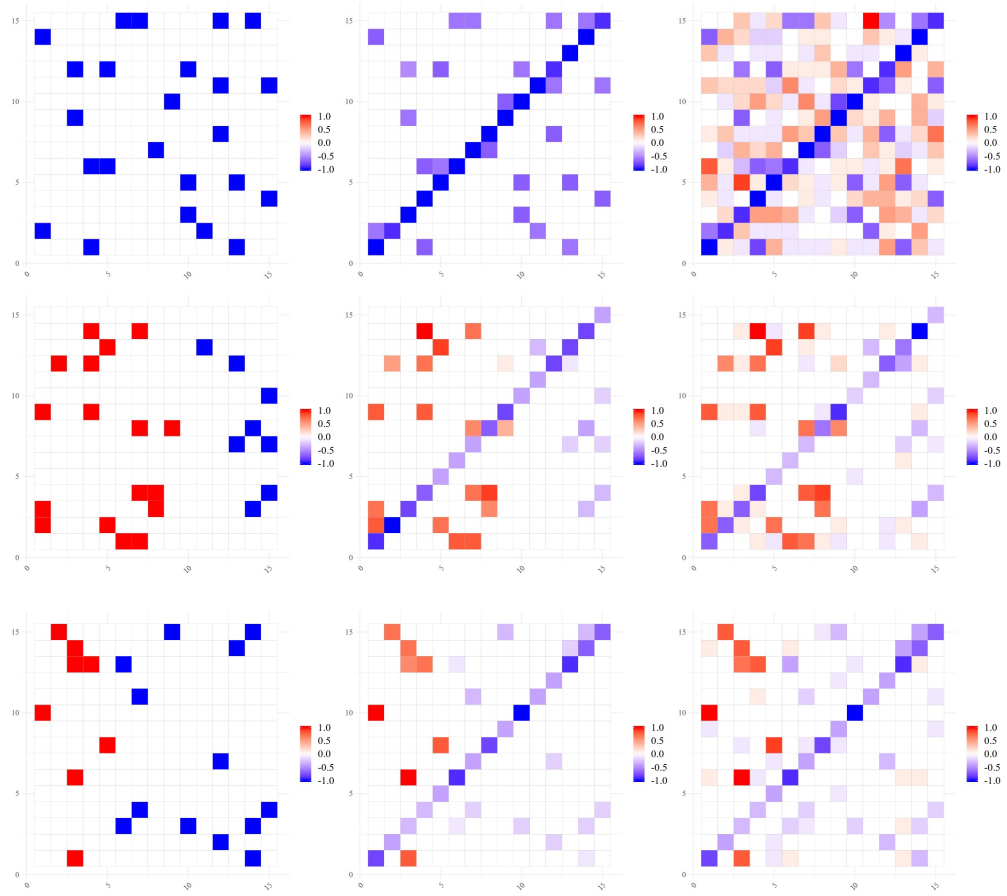

Figure 13: Comparison between LASSO and OLS method. From left to right: original graph, graph reconstructed by LASSO with the minimal counting distance and by OLS. LASSO reconstruction is significantly better (by both counting distance and L^2 distance) than OLS when all the connections are inhibitory. With higher proportion of excitatory neuron, their performances are nearly equivalent in L^2 norm.

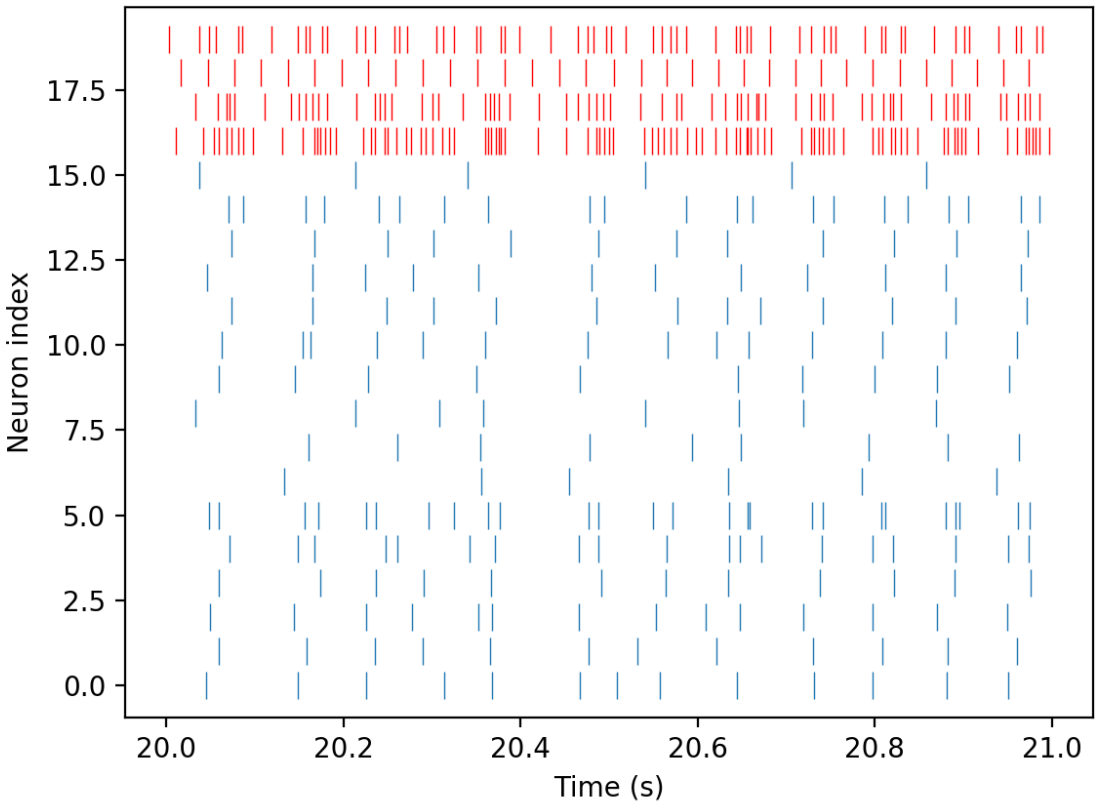

Figure 14: Neurons firing in time-locked patterns under STDP.

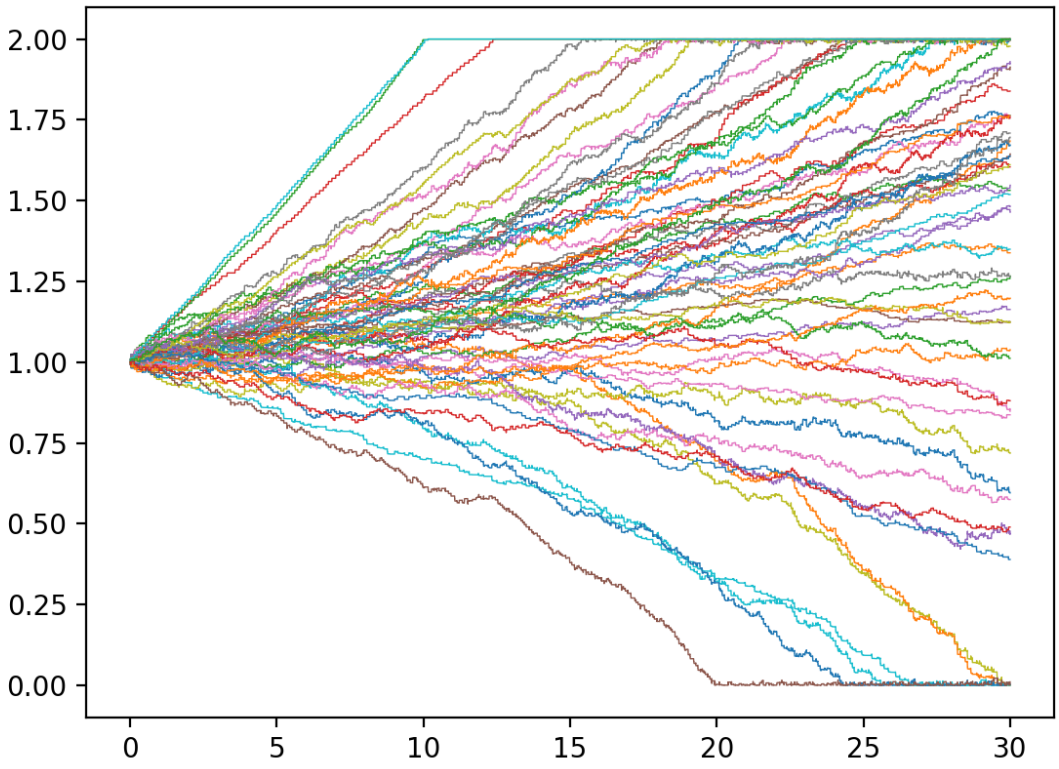

Figure 15: Evolution of synaptic weights under STDP.

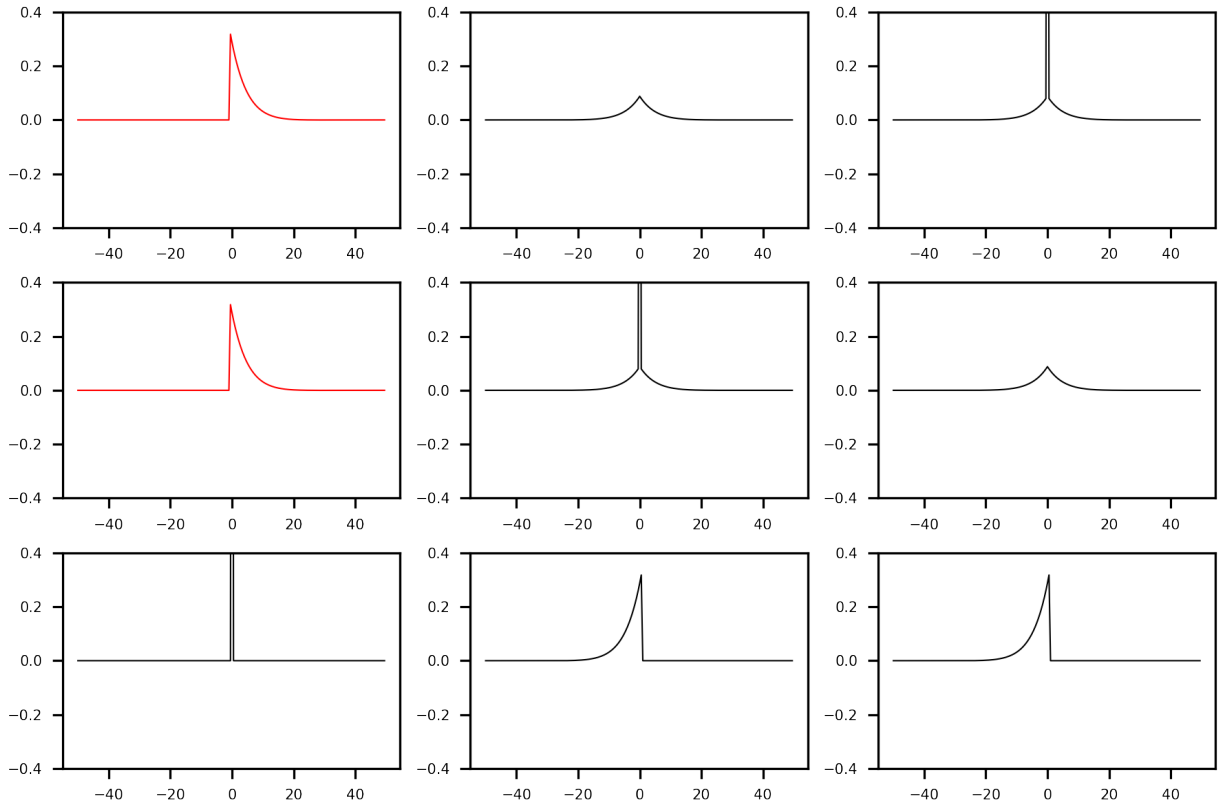

Figure 16: Correlation densities $C_{j\to i}(\tau)$ in a 3-neuron network. $h_{0\to 1}(t) = h_{0\to 2}(t) = C\frac{e^{-t/t_0}}{t_0}$ with $C = 0.5, t_0 = 5ms$ and $h_{j\to i} = 0$ for other connections. $C_{j\to i}$ is drawn at position (j, i). The position $(0, 0)$ corresponds to the bottom left figure. The figures drawn with red lines correspond to $C_{j\to i}$ where $j \to i$ is a network connection.

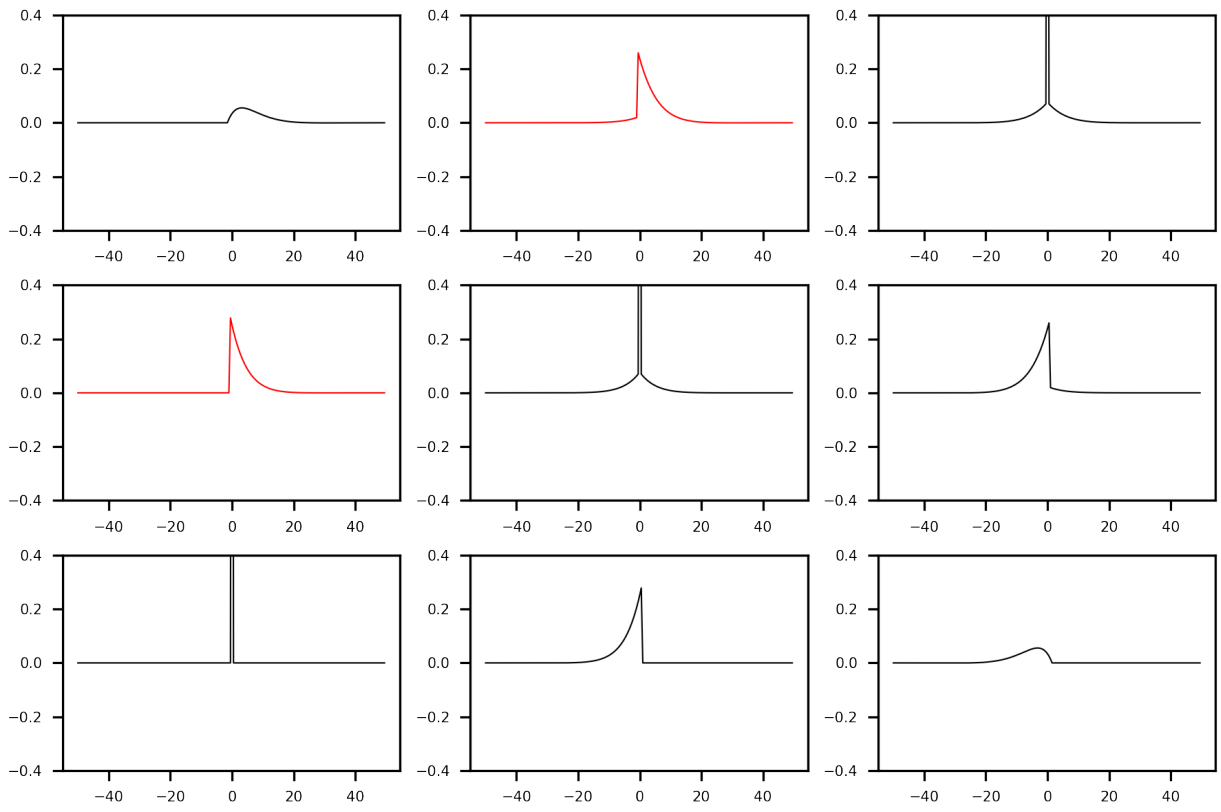

Figure 17: Correlation densities $C_{j\to i}(\tau)$ in a 3-neuron network. $h_{0\to 1}(t) = h_{1\to 2}(t) = C\frac{e^{-t/t_0}}{t_0}$ with $C = 0.5, t_0 = 5ms$ and $h_{j\to i} = 0$ for other connections.

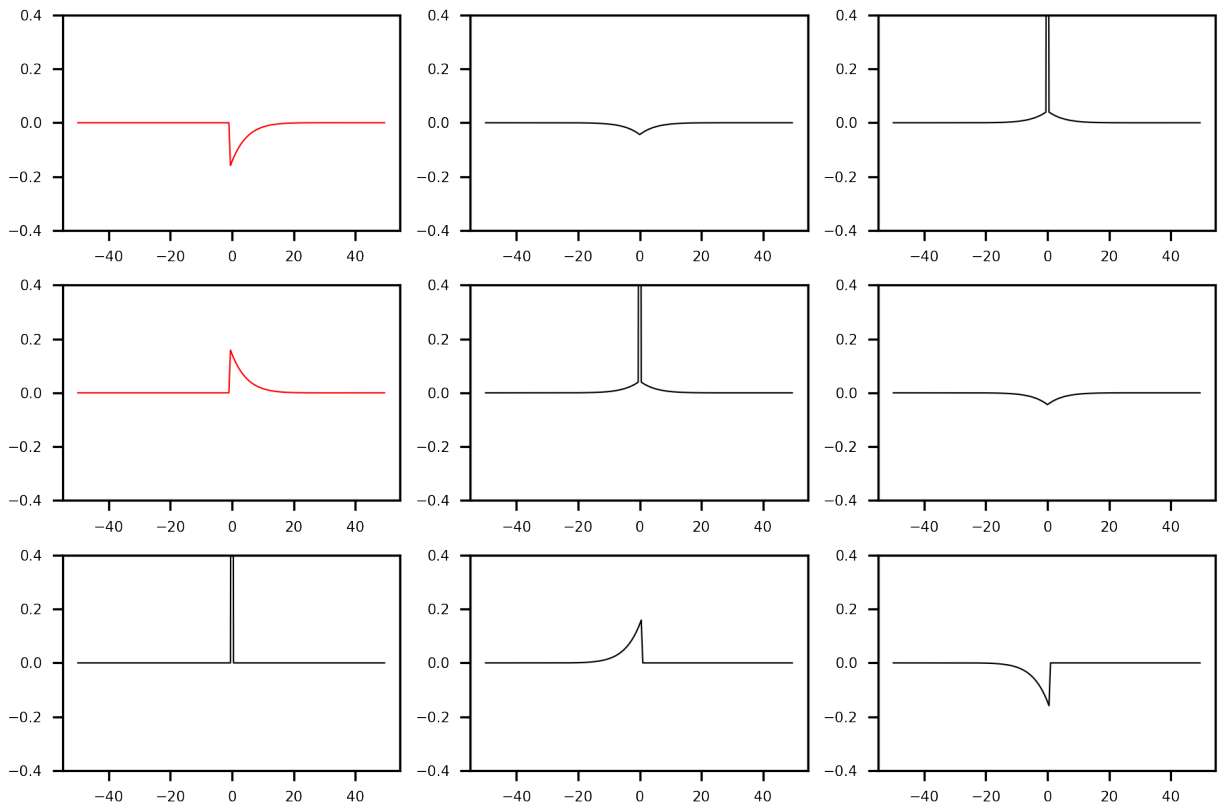

Figure 18: Correlation densities $C_{j\to i}(\tau)$ in a 3-neuron network. $h_{0\to 1}(t) = -h_{0\to 2}(t) = C\frac{e^{-t/t_0}}{t_0}$ with $C = 0.5, t_0 = 5ms$ and $h_{j\to i} = 0$ for other connections.

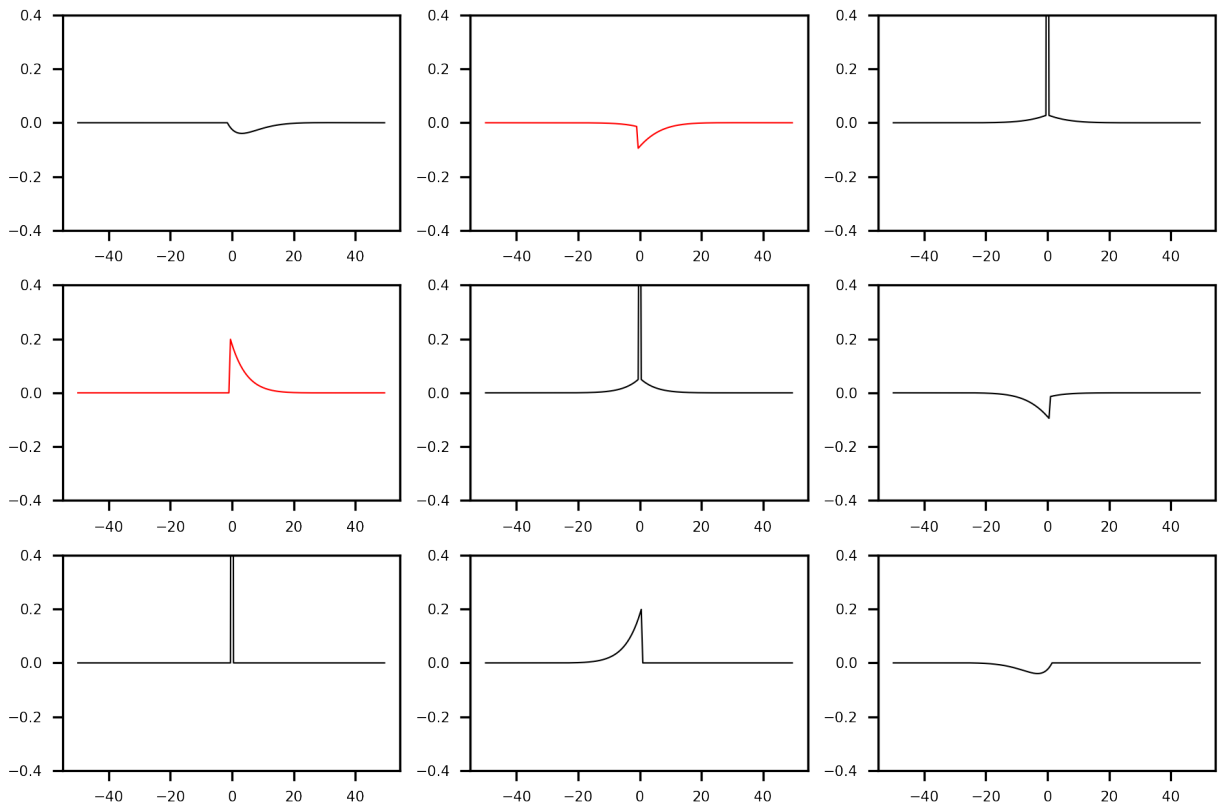

Figure 19: Correlation densities $C_{j\to i}(\tau)$ in a 3-neuron network. $h_{0\to 1}(t) = -h_{1\to 2}(t) = C\frac{e^{-t/t_0}}{t_0}$ with $C = 0.5, t_0 = 5ms$ and $h_{j\to i} = 0$ for other connections.

11 Conclusion

The internship consists of several pieces: neuron models, STDP, information theory, Hawkes process and LASSO method. While we have investigate the LASSO method, simulated a network with STDP, and calculated the correlations in a network, it is still difficult to fit all these pieces together. The goal of estimating how mutual information between spike trains evolve in a plastic neural network is challenging, there are still some intermediate questions that can be answered such as information flow in a non-plastic network and and its relation with network structure.

References

[1] Izhikevich, E. M. (2003). *Simple model of spiking neurons.* IEEE Transactions on Neural Networks, 14(6), 1569–1572.

[2] Izhikevich, E. M., & Desai, N. S. (2003). *Relating STDP to BCM.* Neural Computation, 15(7), 1511–1523.

[3] Lambert, R. C., Tuleau-Malot, C., Bessaih, T., Rivoirard, V., Bouret, Y., Leresche, N., & Reynaud-Bouret, P. (2018). *Reconstructing the functional connectivity of multiple spike trains using Hawkes models.* Journal of Neuroscience Methods, 297, 9–21.

[4] Hansen, N. R., Reynaud-Bouret, P., & Rivoirard, V. (2015). *Lasso and probabilistic inequalities for multivariate point processes.* Bernoulli, 21(1), 83–143.

[5] Pernice, V., Staude, B., Cardanobile, S., & Rotter, S. (2011). *How Structure Determines Correlations in Neuronal Networks.* PLoS Computational Biology, 7(5), e1002059.

[6] Bi, G., & Poo, M. (1998). *Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type.* The Journal of Neuroscience, 18(24), 10464–10472.

[7] Shannon, C. E. (1948). *A Mathematical Theory of Communication.* Bell System Technical Journal, 27(3), 379–423.

[8] Hodgkin, A. L., & Huxley, A. F. (1952). *A quantitative description of membrane current and its application to conduction and excitation in nerve.* The Journal of Physiology, 117(4), 500–544.

[9] Strong, S. P., Koberle, R., de Ruyter van Steveninck, R. R., & Bialek, W. (1998). *Entropy and Information in Neural Spike Trains.* Physical Review Letters, 80(1), 197–200.

[10] Goodman, D. (2008). *Brian: a simulator for spiking neural networks in Python.* Frontiers in Neuroinformatics, 2.

[11] Hawkes, A. G. (1971). *Point Spectra of Some Mutually Exciting Point Processes.* Journal of the Royal Statistical Society: Series B (Methodological), 33(3), 438–443.

[12] Hawkes, A. G. (1971). *Spectra of some self-exciting and mutually exciting point processes.* Biometrika, 58(1), 83–90.

[13] Gerstner, Wulfram, et al. *Neuronal Dynamics from Single Neurons to Networks and Models of Cognition.* Cambridge University Press, 2016.

[14] Cessac, B. (2008). *On Dynamics of Integrate-and-Fire Neural Networks with Conductance Based Synapses.* Frontiers in Computational Neuroscience, 2, 1–20.

[15] B.P. Zeigler, A. Muzy, E. Kofman (2018) *Theory of Modeling and Simulation - Discrete Event and Iterative System Computational Foundations* 3rd edition, Springer, Academic Press

[16] Izhikevich, E. M. (2006). *Polychronization: Computation with Spikes.* Neural Computation, 18(2), 245–282.

[17] Erdös, P., and Rényi, *On random graphs.* Publicationes Mathematica 6, 290–297.

[18] Jin, X., Rast, A., Galluppi, F., Davies, S., Furber, S. (2010). *Implementing spike-timing-dependent plasticity on SpiNNaker neuromorphic hardware.* The 2010 International Joint Conference on Neural Networks (IJCNN).

[19] Strong SP, de Ruyter van Steveninck RR, Bialek W, Koberle R. *On the application of information theory to neural spike trains.* Pac Symp Biocomput. 1998:621-32.