Brain-Inspired Learning Machines

Mathematical Models of Biological Neurons

Emre Neftci

Department of Cognitive Sciences, UC Irvine,

October 3, 2016

Why take inspiration from neurons and the brain? Brain combines many desirable features:

- Resilience and Fault Tolerance,

Why take inspiration from neurons and the brain? Brain combines many desirable features:

- Resilience and Fault Tolerance,
- Adaptability,

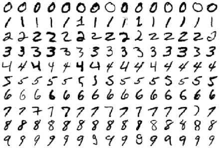
Tenenbaum, Kemp, Griffiths, and Goodman, *science*, 2011

Why take inspiration from neurons and the brain? Brain combines many desirable features:

- Resilience and Fault Tolerance,
- Adaptability,

Tenenbaum, Kemp, Griffiths, and Goodman, *science*, 2011

- Robustness to noisy and ambiguous information,



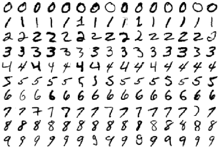
The Brain as a Model of an Artificial Mind

Why take inspiration from neurons and the brain? Brain combines many desirable features:

- Resilience and Fault Tolerance,
- Adaptability,

Tenenbaum, Kemp, Griffiths, and Goodman, *science*, 2011

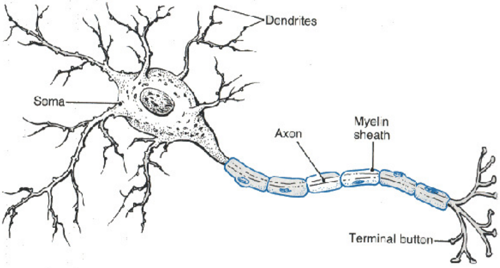
- Robustness to noisy and ambiguous information,



- Power efficiency.

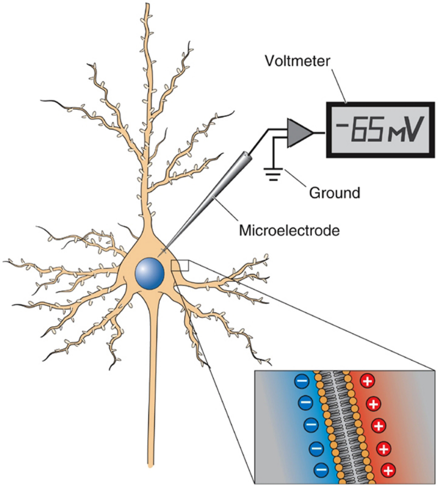
The brain outperforms the computer on almost every task (except arithmetics)

Modeling of Neural Networks

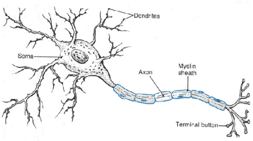


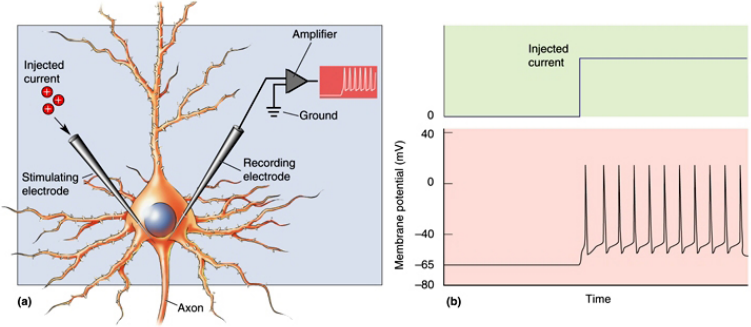
- Dendrites: act as inputs ports
- Soma: the body of the cell, usually where inputs converge and where action potentials are generated
- Axon: propagates action potentials along to other neurons
- Terminal Boutons (Synapses): act as outputs of the neuron

Membrane potential

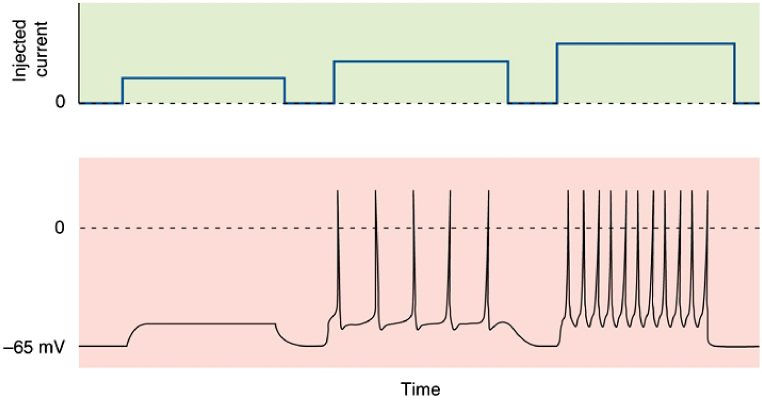


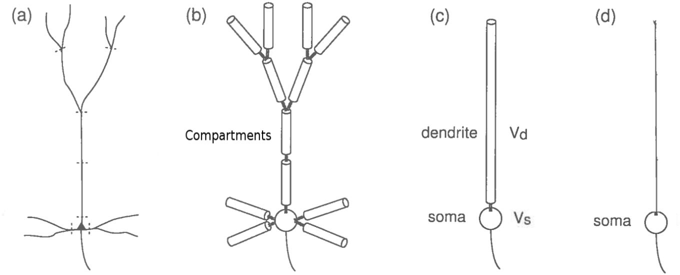
Action Potentials and the Axon



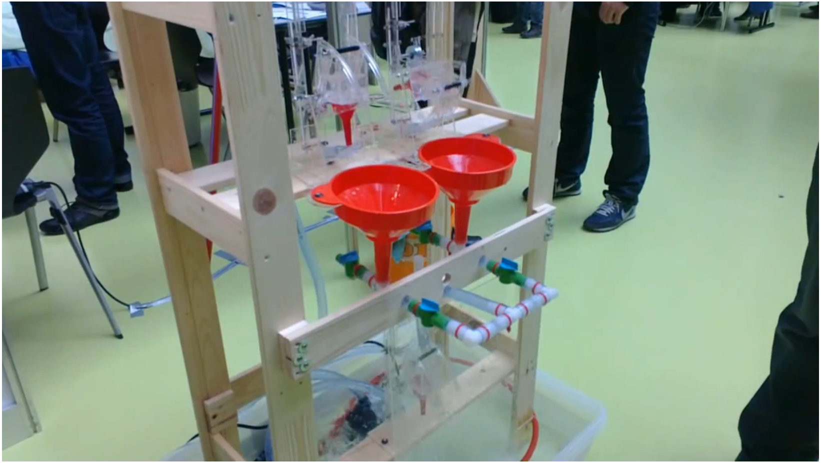


Neurons communicate by all-or-none events called Action Potentials, or "Spikes"


Injected current
0
0
–65 mV
Time

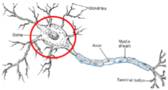


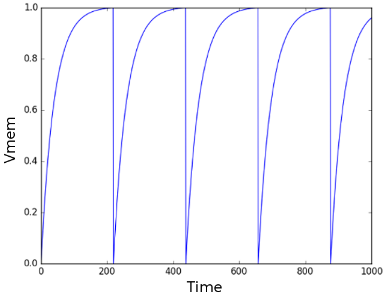
For simplicity, computational neuroscientists often approximate neuron dynamics with one or more compartments "point"

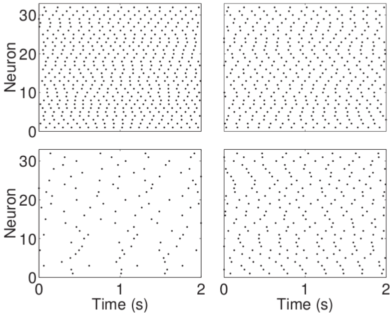


Zurich Brainfair 2012

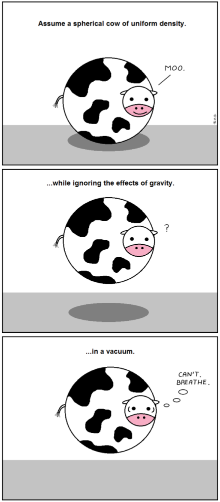
The Leaky Integrate & Fire (IF) Neuron





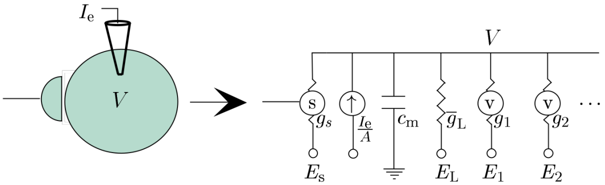


- Inputs are integrated (summed) at the soma.
- Fire and reset: After reaching the threshold, the neuron emits a spike (fires) and is reset
- “Absolute refractory period” after it has spiked (the neuron is silent).



Everything should be made as simple as possible, but not simpler.

A. Einstein



Dayan and Abbott,, 2001

Module (Library) for simulating spiking neurons

Brian2 Neural Simulator

http://briansimulator.org/

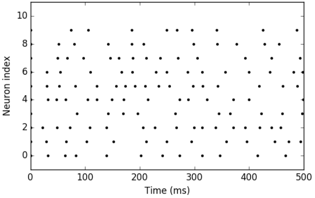
Brette, Goodman, Stimberg, 2016

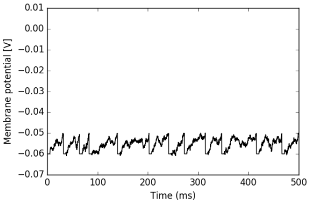
code/import_brian2.py

```
#start ipython and type
from brian2 import *
```

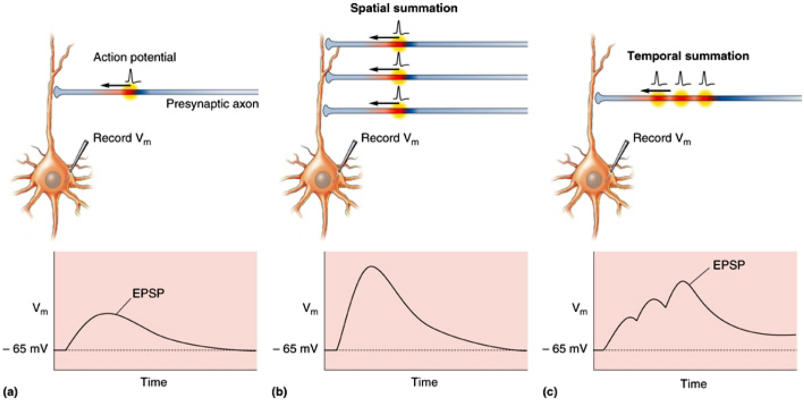
code/brian2_lif.py

```
#Modified from Brian2 documentation examples
from brian2 import *

taum = 20*ms
Vt = -50*mV
Vr = -60*mV
El = -49*mV

eqs = '''
dv/dt = -(v-El)/taum + xi/sqrt(ms)*mV : volt (unless refractory)
'''

P = NeuronGroup(10, eqs, threshold='v>Vt', reset='v = Vr',
                refractory=5*ms, method='euler')

#Monitor spikes
s_mon = SpikeMonitor(P)

#Monitor membrane potential
v_mon = StateMonitor(P, variables='v', record = [0])

run(.5 * second)
```

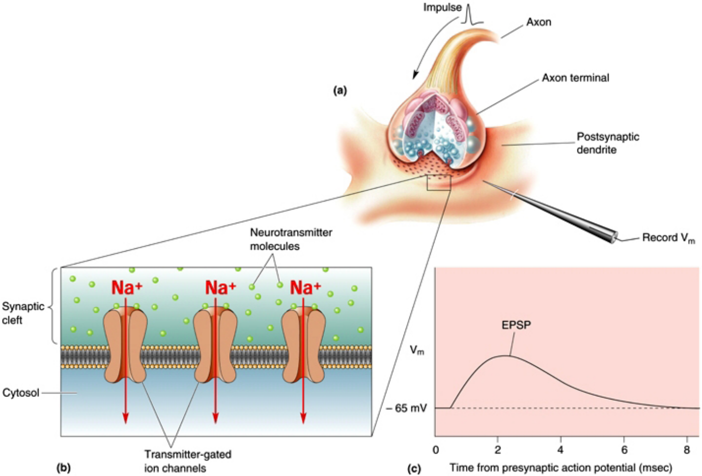




Synapses and Dendrites

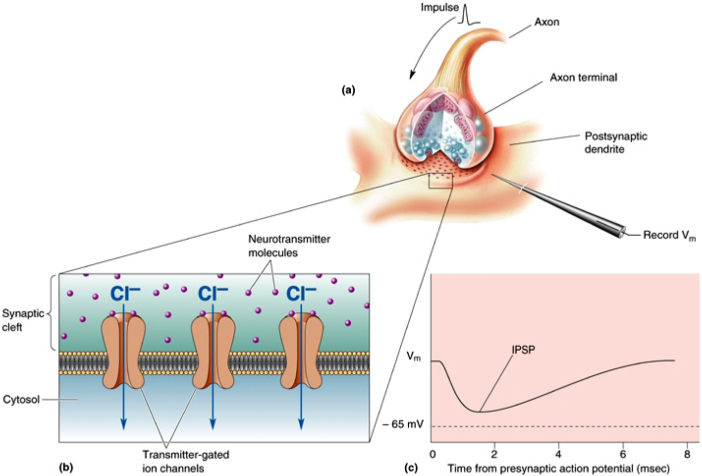


Excitatory Synapses



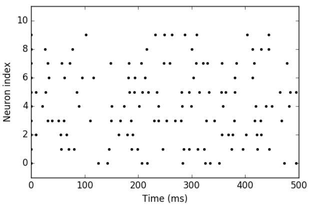
Inhibitory Synapses

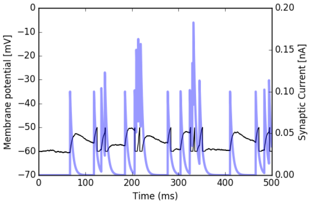
Synapses can also inhibit the target neuron (= depolarize)



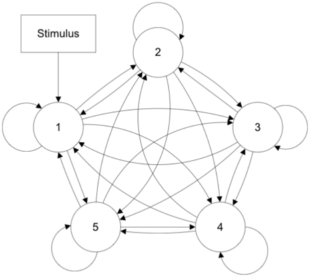
code/brian2_syn.py

```
#Modified from Brian2 documentation examples
from brian2 import *

Cm = 50*pF; gl = 1e-9*siemens; taus = 5*ms
Vt = -50*mV; Vr = -60*mV; El = -60*mV

eqs = '''
dv/dt = -gl*(v-El)/Cm + .1*xi/sqrt(ms)*mV+isyn/Cm : volt (unle
disyn/dt = -isyn/taus : amp
'''

Pin = PoissonGroup(10, rates = 50*Hz)
P = NeuronGroup(10, eqs, threshold='v>Vt', reset='v = Vr',
                refractory=5*ms, method='euler')

w = .1 * nA
S = Synapses(Pin, P, on_pre='isyn += w')
S.connect('i==j')

s_mon = SpikeMonitor(P)

v_mon = StateMonitor(P, variables='v', record = [0])
isyn_mon = StateMonitor(P, variables='isyn', record = [0])

run(.5 * second)

figure(figsize=(6,4))
plot(s_mon.t/ms, s_mon.i, '.k')
```





Neurons can connect to each other



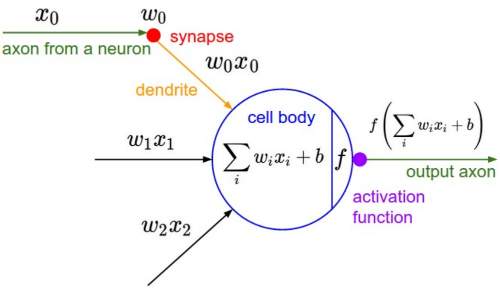
Connections that feed back to the network are called *recurrent*. Others are called *feed-forward*.

In the brain, most of a neuron's input is recurrent:

> *In layer 4 of the cat primary visual cortex [...] only ∼5% of the excitatory synapses arose from the lateral geniculate nucleus (LGN).*

Binzegger, Douglas, and Martin, *J. Neurosci.*, 2004

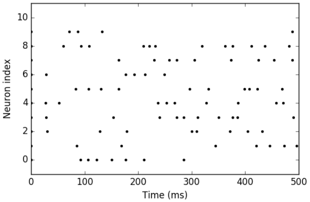
Implementing recurrent connections for neuron i

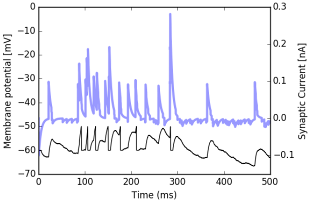


where w_i are the components of a *weight matrix* W.

code/brian2_snn.py

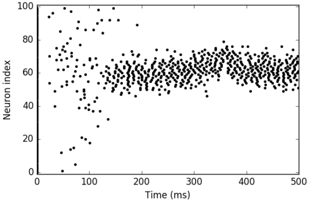
```
from brian2 import *

Cm = 50*pF; gl = 1e-9*siemens; taus = 5*ms
Vt = -50*mV; Vr = -60*mV; El = -60*mV

eqs = '''
dv/dt = -gl*(v-El)/Cm + .1*xi/sqrt(ms)*mV+isyn/Cm : volt (unle
disyn/dt = -isyn/taus : amp
'''

Pin = PoissonGroup(10, rates = 50*Hz)
P = NeuronGroup(10, eqs, threshold='v>Vt', reset='v = Vr',
                refractory=5*ms, method='euler')

wff = .1 * nA
Sff = Synapses(Pin, P, on_pre='isyn += wff')
Sff.connect('i==j')

wrec = -.01 * nA
Srec = Synapses(P, P, on_pre='isyn += wrec')
Srec.connect()

s_mon = SpikeMonitor(P)

v_mon = StateMonitor(P, variables='v', record = [0])
isyn_mon = StateMonitor(P, variables='isyn', record = [0])

run(.5 * second)
```

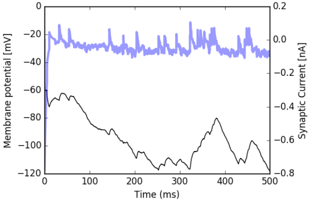




code/brian2_snn_wta.py

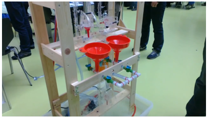
```python
Pin = PoissonGroup(100, rates = 50*Hz)
P = NeuronGroup(100, eqs, threshold='v>Vt', reset='v = Vr',
                refractory=5*ms, method='euler')

wff = .1 * nA
Sff = Synapses(Pin, P, on_pre='isyn += wff')
Sff.connect('i==j')

wreci = -.01 * nA
Sreci = Synapses(P, P, on_pre='isyn += wreci')
Sreci.connect()

wrece = .02 * nA
Srece = Synapses(P, P, on_pre='isyn += wrece')
Srece.connect(condition='abs(i-j)<=10')
```

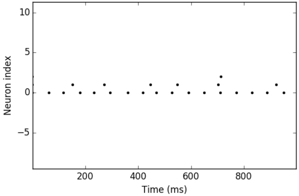


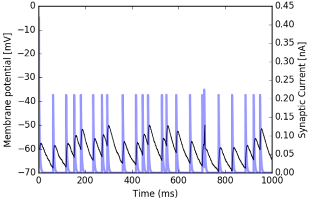


code/brian2_snn_coincidence.py

```
from brian2 import *

Cm = 50*pF; gl = 1e-9*siemens; taus = 3*ms
Vt = -50*mV; Vr = -60*mV; El = -60*mV

eqs = '''
dv/dt = - gl*(v-El)/Cm
        + .1*xi/sqrt(ms)*mV
        + (isyn+iext)/Cm : volt (unless refractory)
disyn/dt = -isyn/taus : amp
iext : amp
'''

P = NeuronGroup(3, eqs, threshold='v>Vt', reset='v = Vr',
                refractory=5*ms, method='euler')
P[0:1].iext = .015 * nA
P[1:2].iext = .01 * nA
P[2:3].iext = -.014 * nA

wrec = .21 * nA
Srec = Synapses(P, P, on_pre='isyn += wrec')
Srec.connect(i=0,j=2)
Srec.connect(i=1,j=2)
```







Integrate-and-fire Neuron Dynamics

$$C_m \frac{\mathrm{d}}{\mathrm{d}t} V_m = -g_L V + I_{syn} + I_{ext} + noise$$

$$\text{if } V_m > V_t : \text{ elicit spike,} \qquad (1)$$

$$\text{and } V_m \leftarrow V_r \text{ during } \tau_{arp}$$

When V_m reaches V_t, an action potential (spike) is elicited. After this, the membrane potential is clamped to the reset potential V_r for a refractory period τ_{arp}.

Current-based Synapse Dynamics

$$\tau_s \frac{\mathrm{d}}{\mathrm{d}t} I_{syn} = -I_{syn} \qquad (2)$$

$$\text{on presynaptic spike} : I_{syn} \leftarrow I_{syn} + w$$

Each incoming spike results in a post-synaptic current that rises *instantly* and decays exponentially

- V_m: Membrane potential
- C_m: Membrane capacitance
- g_L: Leak conductance
- V_r: Reset Potential
- V_t: Firing Threshold
- I_{syn}: Synaptic current
- I_{ext}: External synaptic input
- $\tau_m = C_m/g_L$: Membrane time constant
- τ_s: Synaptic time constant (rate of channel closing)
- τ_{arp}: Absolute Refractory Period
- w: Synaptic Weight

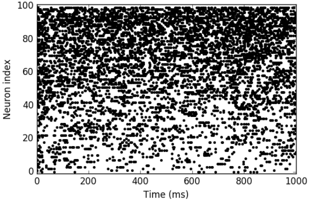
For more information about integrate-and-fire neurons and conductance-based synapses, refer to Dayan and Abbott book, Section 5.4 and 5.8

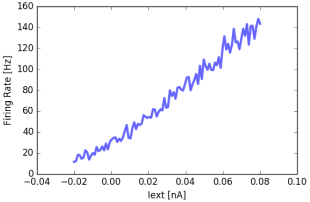
Activation Function $\phi(I)$: Firing rate of a spiking neuron as a function of input drive I.

Average Firing Rate - Neuron Activation Function

code/brian2_activation_function.py

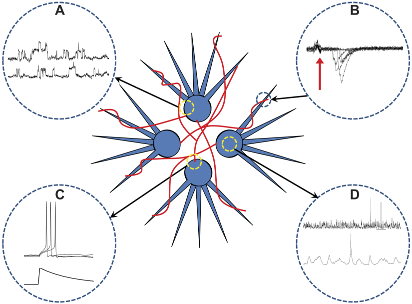
```
def mean_rates(s_mon):
    rates = np.zeros(np.max(s_mon.i)+1)
    for k,v in s_mon.spike_trains().iteritems():
        rates[k] = len(v)
    return rates/s_mon.t[-1]
#Modified from Brian2 documentation examples
from brian2 import *

Cm = 50*pF; gl = 1e-9*siemens; taus = 5*ms
Vt = 10*mV; Vr = 0*mV; El = 0*mV
sigma = 5./sqrt(ms)
eqs = '''
dv/dt = -gl*(v-El)/Cm
        + sigma*xi*mV
        +iext/Cm : volt (unless refractory)
iext : amp
'''

P = NeuronGroup(100, eqs, threshold='v>Vt', reset='v = Vr',
                refractory=0*ms, method='milstein')

P.v = Vr
P.iext = np.linspace(-.2, .8, len(P))*.1*nA

s_mon = SpikeMonitor(P)

run(5.0 * second)
```





Steady state activity in the Winner-Take-All Network

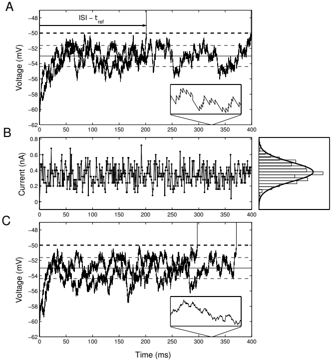
Mean-field Models for Stochastic Spiking Neurons



Yarom and Hounsgaard, *Physiological Reviews*, 2011

Fusi and Mattia, *Neural Computation*, 1999

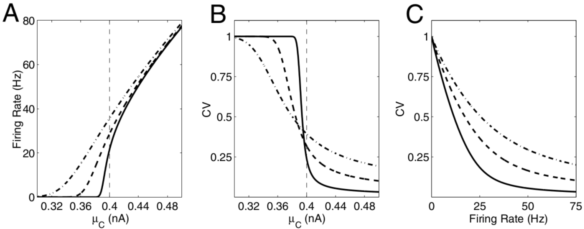
Diffusion Approximation



$$\nu(I) = \left(\tau_{ARP} + \tau_m \sqrt{\pi} \int_{\frac{V_r - V_0}{\sigma_V}}^{\frac{\theta - V_0}{\sigma_V}} \exp(x^2)(1 + \mathrm{erf}(x))\mathrm{d}x \right)^{-1}$$

$$V_0 = \frac{I}{g_L}$$

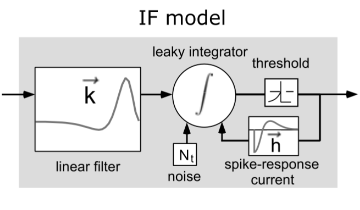
$$\sigma_V = \sqrt{\frac{\sigma^2}{g_L C_m}} \tag{3}$$



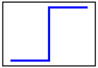
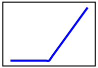
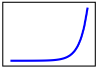
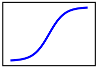
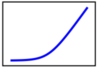
Probability that a neuron with absolute refractory period spikes

$$P(\text{spike at time } t|\text{spiked at time } t') = \begin{cases} 0 & \text{if } t - t' < \tau_{ARP} \\ \nu(I, \tau_{ARP} = 0) & \text{if } t - t' \geq \tau_{ARP} \end{cases}$$

Linear, Non-Linear, Poisson (LNP) Neuron



Firing Rate of a Neuron: Commonly Used Activation Functions

Name		
Threshold	$\theta(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0 \end{cases}$	
Rectified Linear	$[x]_+ = \max(x, 0)$	
Exponential	$f(x) = \exp(x)$	
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	
Soft plus	$S(x) = \log(1 + e^x)$	
Soft plus with ARP	$S_{RP}(x) = \frac{S(x)}{S(x)+1}$