

Neurasmus, Amsterdam, July 4th 2024 Mathematical Models of Neural Activity

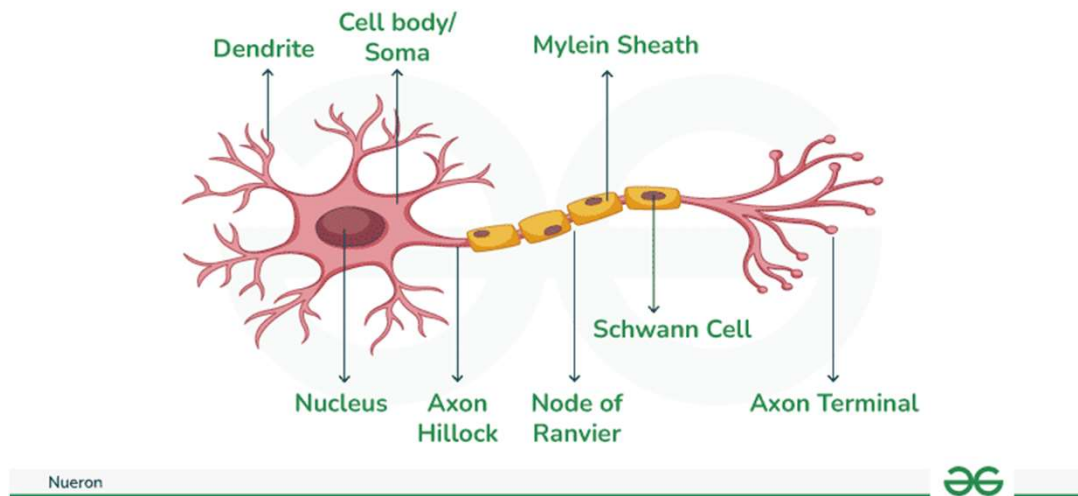
NicolasDoyon

Laval University, Québec Canada



Introduction and biophysical principles

What do neurons do?



- Neurons **receive information** through their dendrites,
- They **integrate information** in their soma,
- They **transmit information** to other neurons through their axon.

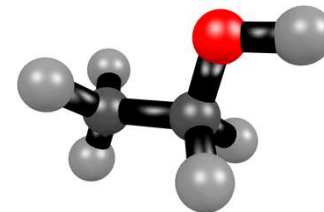
Different types of models

- According to the level of details we want to include, there are several possible modeling approaches.
- A more complex or more complete model is not always better.



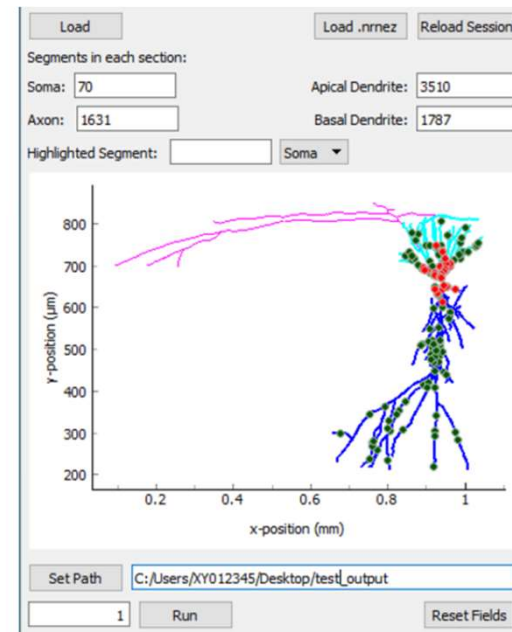
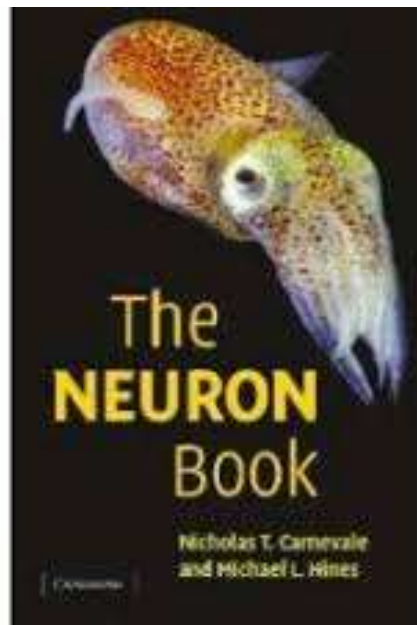
Scale matters

(single protein, synapse,
single cell, network)



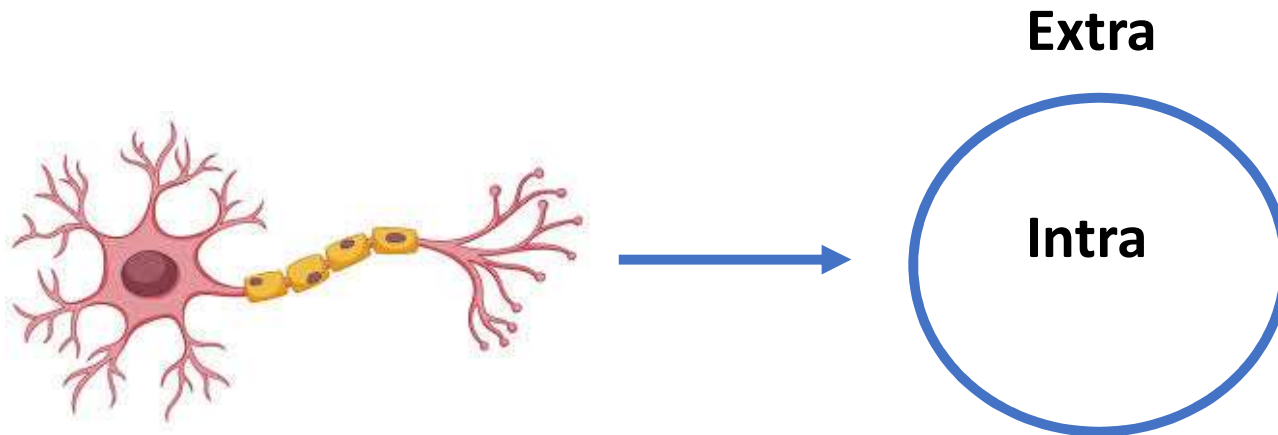
The NEURON software

- For those who want to know more about detailed single cell models, the **NEURON software** is tailor made to develop modular models of single neurons. <https://modeldb.science/>



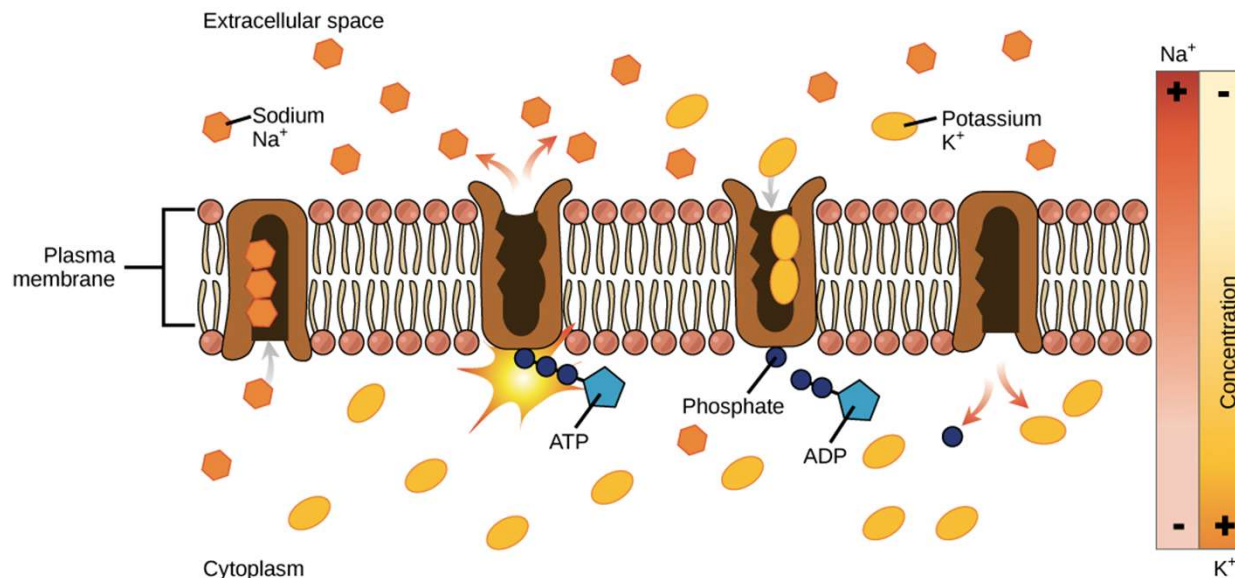
Single compartment models

We will neglect the spatial dimension of the neuron and treat the intracellular space as a single homogeneous compartment.



Some biophysical principles

- The membrane acts as a separation between the intracellular and extracellular medium.
- Intra and extracellular concentrations are different.
- This difference is maintained by the sodium potassium pump which uses ATP to transport ions against their concentration gradient.



Typical ionic concentrations

Ion type	Intra	Extra
chloride	10 mM	140 mM
Sodium	15 mM	120 mM
Potassium	140 mM	3 mM
Calcium	100 nM	1 mM

<https://www.khanacademy.org/science/ap-biology/cell-structure-and-function/facilitated-diffusion/a/active-transport>

The Nernst potential

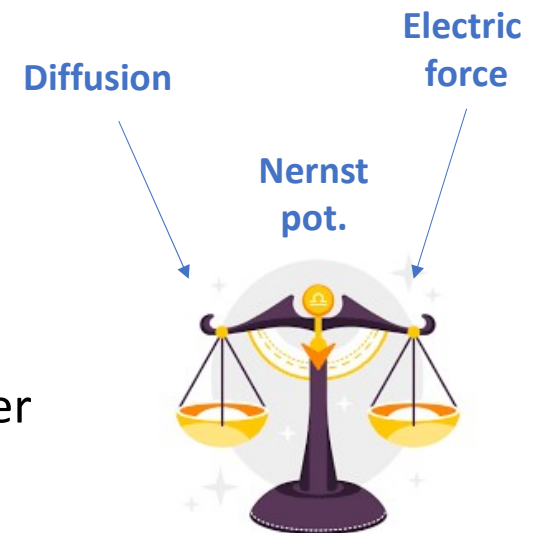
Ion gradients =



Two forces regulate ion movement across the membrane.

1. Diffusion (technically not a force)
2. Electric force

The **Nernst reversal potential** of an ion is the value of the membrane potential for which these two forces balance each other out.



The Nernst potential

The **Nernst equation** is given by:

$$E_x = \frac{RT}{zF} \log \left(\frac{[x]_o}{[x]_i} \right).$$

- $R=8.314$, perfect gas constant,
- $T=310$, temperature in Kelvin,
- $F=96\,845$, Faraday constant,
- z , valence of the ion,
- $[x]_o$, extracellular concentration,
- $[x]_i$, intracellular concentration
- log in base e , E_x in volts.

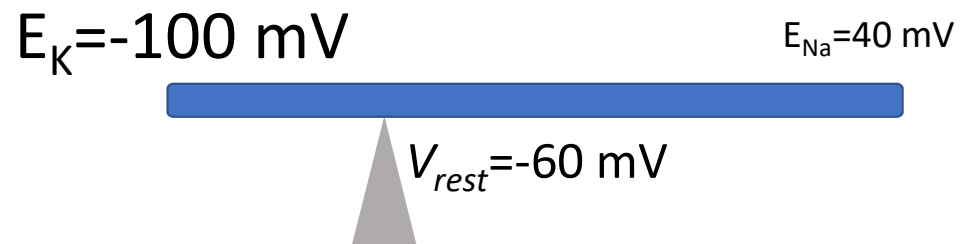
Ion x	E_x
Na^+	40 mV
K^+	-100 mV
Cl^-	-60 mV

For **potassium** for example, we have

$$E_K = \frac{RT}{F} \log \left(\frac{3}{140} \right) \approx -100 \text{ mV}.$$

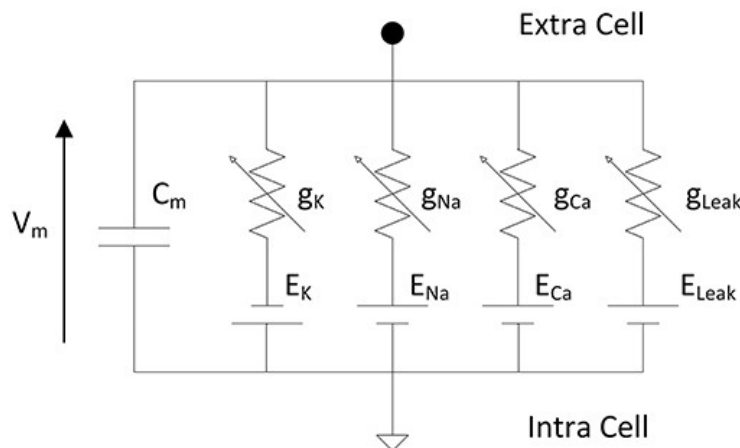
When $V_m = -100 \text{ mV}$:

- The negative potential attracts potassium (K^+) inside the cell.
- This balances the diffusion that drives K^+ out.



Neurons as electrical circuits

Neurons are often schematized as an **electrical circuit**.



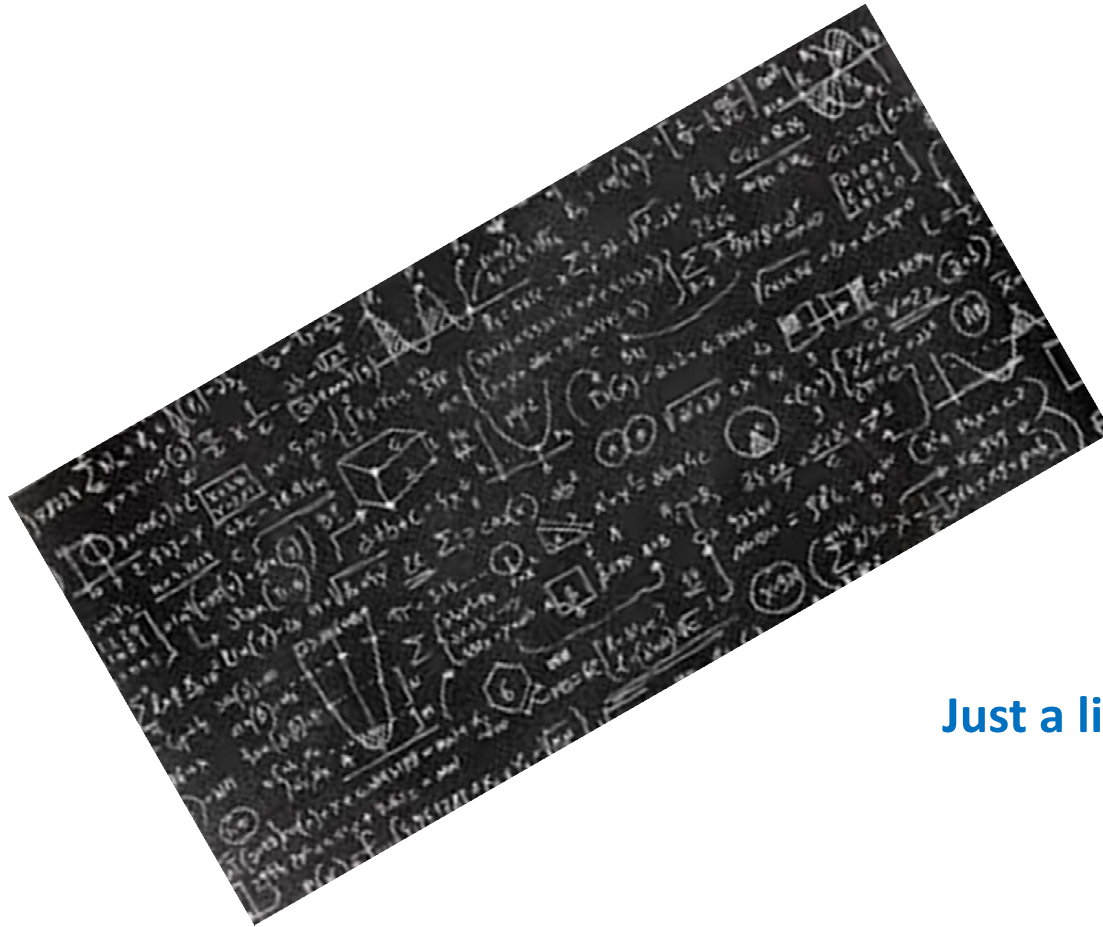
<https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2019.00377/full>

- The membrane is an electrical insulator. It acts as a capacitor. As it is about 5 nm wide, the electrical capacity is very small. So the cell is always close to electroneutrality.

$$\sum z_x [x]_i \approx 0.$$

- Ionic gradients make batteries.
- Conductances change with time as channels open and close.
- Ultimately, the ATP to ADP reaction is charging the batteries.

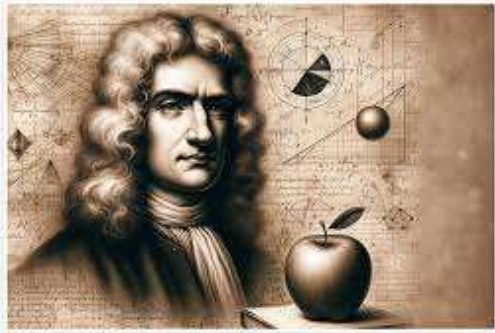
Now some math...



Just a little bit...

Ordinary Differential Equations (ODEs)

Ordinary differential equations are a powerful mathematical tool to describe variables that evolve **continuously** with time.



If x is the height of a falling apple:

$$x'' = -g.$$

The general form of an ode is given by: $\frac{dx}{dt} = f(x, t)$.

Here x can be a single variable or a vector describing several variables.

Ordinary differential equations


- ODEs describe the *rate of change* of certain quantities. To obtain solutions, we always have to specify the initial state of the system (*initial conditions*).



- It is very rarely possible find an exact solution to ODEs.

- We have to use *numerical methods*.



- Don't worry, **python**  (or any language, does it for you).

- In Python, there are several options, **odeint** is a simple one.

Ordinary differential equations

`scipy.integrate.`
odeint

You have to
importhe the
package

```
graph TD; A[You have to importhe the package] --> B[scipy.integrate.odeint]; B --> C[The equation you want to solve]; B --> D[Initial conditions]; B --> E[A list of the time points on wich you want the solution]; B --> F[Parameters such as conductances];
```

odeint(*func*, *y0*, *t*, *args=()*)

The equation
you want to
solve

Initial
conditions

A list of the
time points
on wich you
want the
solution

Parameters
such as
conductances

Ordinary differential equations

How does the computer do it?

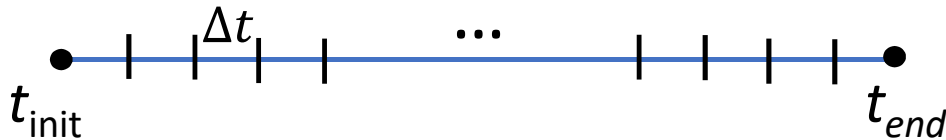
Some integration methods can be quite complicated...

We present the *Euler forward* method.

Downside: Imprecise.

Upside: Simple we can implement it manually in a for loop when odeint cannot be used.

We divide the time interval in steps of length Δt



We compute on each time step: $x' = f(x) \rightarrow x(t + \Delta t) \approx x + \Delta t f(x)$.

Ordinary differential equations

Example: Suppose you want to solve
 $x'(t) = x, x(0) = 1, 0 \leq t \leq 4.$

For $\Delta t = 1$, we have

$$x(1) \approx x(0) + x'(0) = 2,$$

$$x(2) \approx x(1) + x'(1) = 4,$$

$$x(3) \approx x(2) + x'(2) = 8,$$

$$x(4) \approx x(3) + x'(3) = 16.$$

For $\Delta t = 0.5$, we have

$$x(0.5) \approx x(0) + 0.5 x'(0) = 1.5,$$

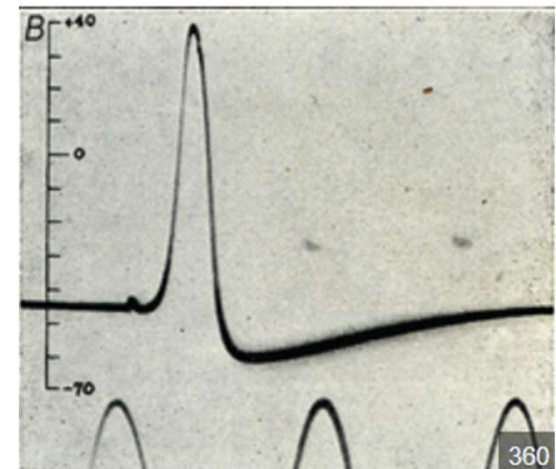
$$x(1) \approx x(0.5) + 0.5x'(0.5) = 2.25,$$

....

$$x(4) \approx x(3.5) + 0.5x'(3.5) = 25.62.$$

When we choose a smaller Δt , the solutions are more precise but resolution takes longer.

The Hodgkin-Huxley model



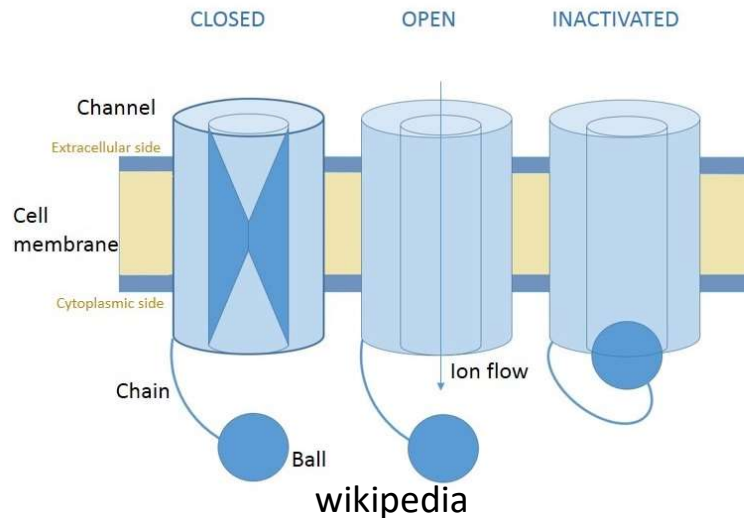
The Hodgkin-Huxley model

- The Hodgkin-Huxley model has an historical and practical importance.
- It has been derived from experiments on the **giant squid axon**.
- The authors won the **Nobel prize in Physiology or Medicine in 1963** for their work.
- Their model rely on the description of voltage gated sodium and potassium channel.

Sodium and potassium voltage gated channels

- The voltage gated potassium and sodium channels are responsible for generating the action potential.
- Gates open and close according to the membrane potential.

Sodium channels have two types of gates, including an **inactivating** one.



Sodium and potassium voltage gated channels

The proportion of open channels sodium is given by

$$m^3 h$$

where h is an inactivation gate.

The proportion of open potassium channels is given by

$$n^4$$

since the opening of potassium channels are regulated by 4 molecular gates.

The presence of exponents may seem surprising. Their values have been empirically derived.

The Hodgkin-Huxley equations

Putting all this together, we get the following system of equations that specify the HH model:

$$\begin{aligned}C \frac{dV}{dt} &= g_L(E_l - V) + \overline{g_{Na}} m^3 h (E_{Na} - V) + \overline{g_K} n^4 (E_K - V) + \mathbf{I}, \\ \frac{dm}{dt} &= \alpha_m(V)(1 - m) - \beta_m(V)m, \\ \frac{dh}{dt} &= \alpha_h(V)(1 - h) - \beta_h(V)h, \\ \frac{dn}{dt} &= \alpha_n(V)(1 - n) - \beta_n(V)n.\end{aligned}$$

The functions α and β stand respectively for the **opening** and **closing** rate functions of the gating variables.

Due to the form of the equations, the gating variables will always remain between 0 and 1.

Qualitative behaviour of the Hodgkin-Huxley model

The parameter I corresponds to the external current injected into the cell. Its value determines the behaviour of the neuron

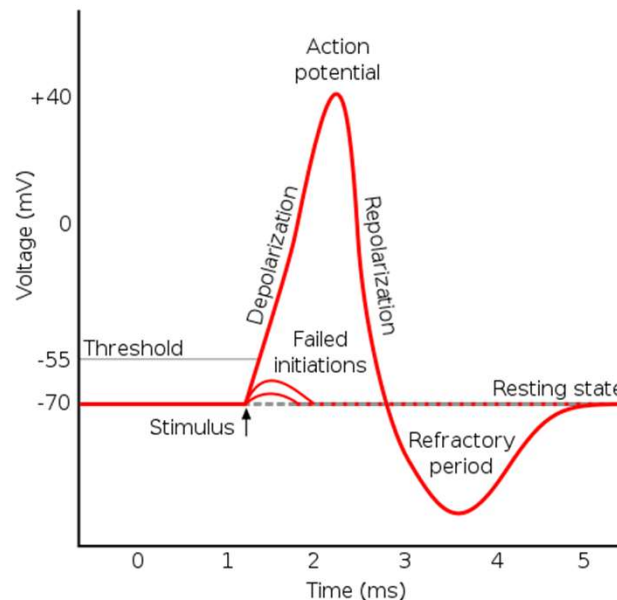
1. If the value of I is small, than the neuron remains at resting state.
2. For intermediate values of I , the neuron emits action potentials.
3. For very strong values of I , the neuron is in an inactive depolarized state.

Stages of an action potential in the HH model

1. The membrane depolarizes.
2. The **m** gates open, sodium enters and membrane further depolarizes (around -30 mV to 20 mV)
3. When the cell is depolarized, the **h** gate closes and the **n** gates opens.
4. Due to potassium currents, the cell hyperpolarizes.
5. The **n** gates close and the **h** gate opens.

Leaky Integrate and Fire model (LIF)

- One possible simplification is to omit the description of the action potential altogether.
- This is interesting from the point of view of model simplicity since the typical shape of an action potential is rather difficult to replicate.



Leaky Integrate and Fire model (LIF)

The equation specifying the **LIF** model is given by:

$$C \frac{dV}{dt} = g_L(E_L - V) + I.$$
$$V \leftarrow V_{reset}, \quad \text{if } V > V_{tresh}$$

The second line is to be interpreted as:

When **V** reaches a threshold value (**V_{tresh}** usually around -30 mV), we assume that an action potential occurs and that the potential is instantaneously reset.

The LIF model is not specified by ODEs.

Since it is discontinuous, we cannot directly use **odeint** to solve it.

We will solve it '*manually*', by updating the value of the membrane potential at each time step in a for loop.

Leaky Integrate and Fire models.

Thanks to their simplicity, **LIF** models are often used to describe **neural networks**.

When used in a network context, the equation describing the evolution of LIF neurons becomes:

$$C \frac{dV_j}{dt} = g_L(E_L - V_j) + \sum_i w_{ji} \delta(t - spike_i) + I_j.$$
$$V_j \leftarrow V_{reset}, \quad \text{if } V_j > V_{tresh}.$$

Here, V_j stands for the membrane potential of neuron j , w_{ji} is the connection weight from neuron j to neuron i , $spike_i$ is the time of occurrence of a spike in neuron i and δ is the Dirac delta.

Wilson-Cowan (WC) models

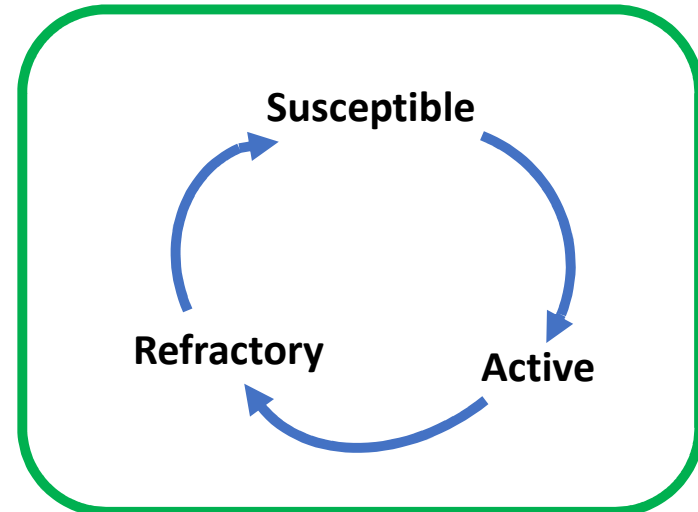
Another approach to simplify models of neural activity is to divide the states of neurons into a finite set of possible states according to their level of activity.

The Wilson-Cowan model divides neurons into 3 states:

Active: The neuron emits an action potential.

Susceptible: Neurons are at resting state but can become active if they receive enough stimulation.

Refractory: Shortly after firing, neurons cannot firing again no matter the input.



Wilson-Cowan model

Remark: Wilson-Cowan models don't describe the state of individual neurons.

$A(t)$: The **proportion** of neurons in active state at time t .

$R(t)$: The **proportion** of neurons in refractory state at time t .

$S(t)$: The **proportion** of neurons in susceptible state at time t .

Transitions:

$A \rightarrow R$ at constant rate,

$R \rightarrow S$ at constant rate,

$S \rightarrow A$ at a rate proportional to $F(\text{input})$
where F is an increasing activation function.

Wilson-Cowan model

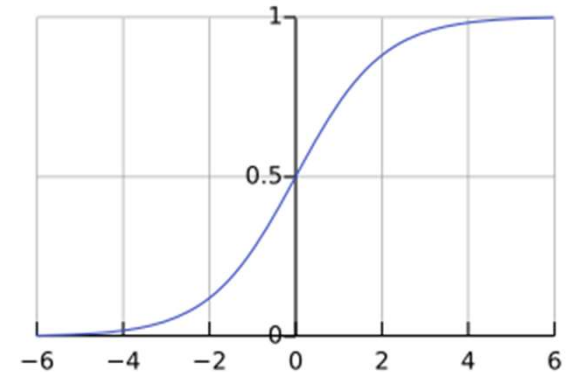
The Wilson Cowan model is specified by the equations:

$$\begin{aligned}\frac{dA}{dt} &= -\alpha A + \beta SF(wA + I), \\ \frac{dS}{dt} &= \gamma R - \beta SF(wA + I), \\ \frac{dR}{dt} &= -\gamma R + \alpha A.\end{aligned}$$

The parameter w can be interpreted as the mean connection weight between two neurons.

The sigmoid function is often used as an activation function

$$\sigma(x) = \frac{1}{1 + e^{-x}}.$$



The Wilson-Cowan model

At first glance, the Wilson-Cowan model has 3 dimension (3 independant variables).

However: $A(t) + S(t) + R(t) = 1$.

Thus, by settting $R(t) = 1 - A(t) - S(t)$ we reduce the model to two variables.

We can also assume that the proportion of refractory neurons is at equilibrium.

$$\frac{dR}{dt} = -\gamma R + \alpha A = 0 \leftrightarrow R = \frac{\alpha}{\gamma} A.$$

With this, we obtain a one dimensional version of the model

$$\frac{dA}{dt} = -\alpha A + \beta \left(1 - \left(1 + \frac{\alpha}{\gamma} \right) A \right) F(wA + I).$$

The Wilson-Cowan model

The **WC** model can also be applied to models with several **populations**.

In this context, a **population** of neurons is an abstract and somewhat arbitrary concept.

A population is a group of neurons with similar properties and expected to have similar levels of activity.

Neurons are often divided into populations of **inhibitory** and **excitatory** neurons. Populations can also be defined according to the spatial positions of neurons.

The equations for **two population WC models** are given by:

$$\frac{dA_1}{dt} = -\alpha A_1 + \beta(1 - A_1)F(w_{11}A_1 + w_{12}A_2 + I_1),$$

$$\frac{dA_2}{dt} = -\alpha A_2 + \beta(1 - A_2)F(w_{21}A_1 + w_{22}A_2 + I_2).$$