Neurons

1. The detector model.
2. Biological properties of the neuron.
3. The computational unit.

To keep it interesting:

- How can this basic processing unit subserve our thoughts, feelings, etc?
- How can we simulate this basic processing unit to explore its role in the above?

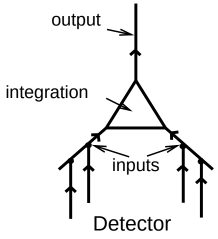
- Also keep in mind this material gets elaborated w/the simulations, and the earliest material is often hardest for those w/primarily psych background.

Detector Model

Each neuron is detecting some set of conditions
(e.g., smoke detector). *Representation* is what is detected.

Neurons feed on each other's outputs — layers of ever more complicated detectors.

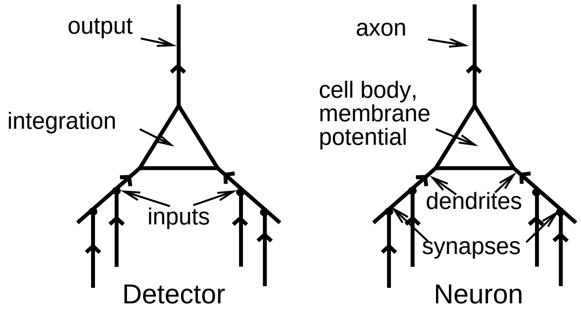
(Things can get very complex in terms of *content*, but each neuron is still carrying out basic detector *function*).



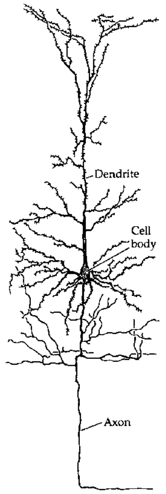
Detector vs. Computer

	Computer	Detector
Memory & Processing	Separate, general-purpose	Integrated, specialized
Operations	Logic, arithmetic	Detection (weighing & accumulating evidence, evaluating, communicating)
Complex Processing	Arbitrary sequences of operations chained together in a program	Highly tuned sequences of detectors stacked upon each other in layers

Understanding Neural Components in Detector Model



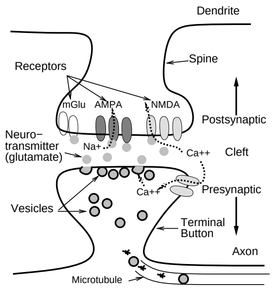
A Real Neuron



Basic Properties of a Neuron

- It’s a cell: body, membrane, nucleus, DNA, RNA, proteins, etc.
- Membrane has **channels**, passing **ions**.
- Cell has electrical **potential**, integrated in cell body, activates **action potential** output in axon, releases **neurotransmitter**.
- Neurotransmitter activates potential via dendritic **synaptic input** channels.
- Excitation and inhibition are transmitted by different neurons.

The Synapse and Synaptic Efficacy



- Presynaptic: # of vesicles released, NT per vesicle, efficacy of reuptake mechanism.
- Postsynaptic: # of receptors, alignment & proximity of release site & receptors, efficacy of channels, geometry of dendrite/spine.

Neurons

- How can this basic processing unit subserve our thoughts, feelings, etc?
- How can we simulate this basic processing unit to explore its role in the above?

Mathematical Abstraction

Ions flow into and out of the neuron under the forces of electricity and concentration gradients (diffusion).

The net result is a electric potential difference between the inside and outside of the cell — **the membrane potential** V_m.

This value represents an integration of the different forces, and an integration of the inputs impinging on the neuron.

We use the equations describing this integration in our models.

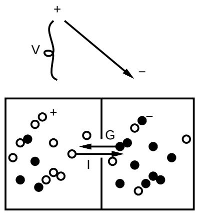
Electricity

Positive and negative **charge** (opposites attract, like repels).

Ions have net charge: Sodium (Na^+), Chloride (Cl^-), Potassium (K^+), and Calcium (Ca^{++}).

Current flows to even out distribution of + and - ions.

Disparity in charges produces **potential** (the potential to generate current).



Resistance

Ions encounter **resistance** when they move.
Neurons have **channels** that limit flow of ions in/out of cell.

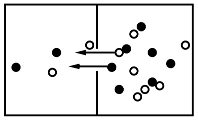
The smaller the channel, the higher the resistance, the greater the potential needed to generate given amount of current (Ohm's law):

$$I = \frac{V}{R}$$

Conductance
G = 1/R, so $I = GV$

Diffusion

Constant motion causes mixing – evens out distribution.



same deal with potentials, conductance, etc:

$$I = -DC$$

(Fick's First law)

Equilibrium

Balance between electricity and diffusion:

E = **Equilibrium** potential = amount of electrical potential needed to counteract diffusion:

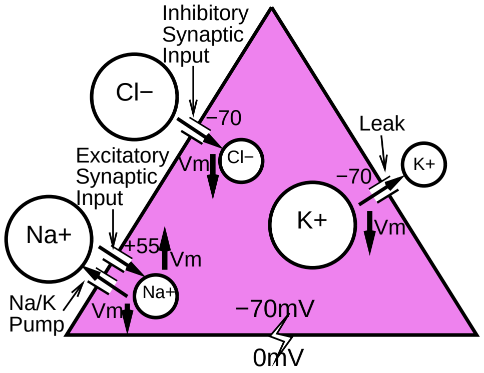
$$I = G(V - E)$$

Also called the:

Reversal potential (because current reverses on either side of E)

Driving potential (because flow of ions drives potential toward this value)

The Neuron and its Ions



Everything follows from the sodium pump, which creates the "dynamic tension" (compressing the spring, winding the clock) for subsequent neural action.

Putting it Together

$$I_c = g_c(t)\bar{g}_c(V_m - E_c) \tag{1}$$

e = excitation (Na^+)
i = inhibition (Cl^-)
l = leak (K^+).

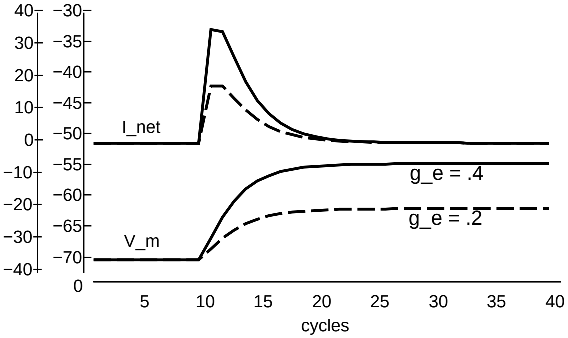
$$\begin{aligned} I_{net} \; &= \; g_e(t)\bar{g}_e(V_m - E_e) + \\ &\quad g_i(t)\bar{g}_i(V_m - E_i) + \\ &\quad g_l(t)\bar{g}_l(V_m - E_l) \end{aligned} \tag{2}$$

$$V_m(t+1) = V_m(t) - dt_{vm} I_{net} \tag{3}$$

or

$$V_m(t+1) = V_m(t) + dt_{vm} I_{net-} \tag{4}$$

In Action



(Two excitatory inputs at time 10, of conductances .4 and .2)

Overall Equilibrium Potential

If you run V_m update equations with steady inputs, neuron settles to new *equilibrium potential*.

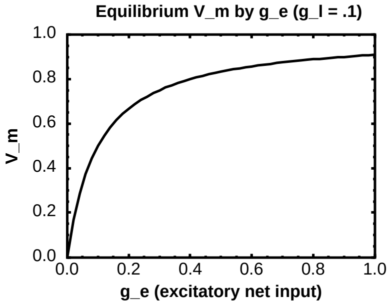
To find, set $I_{net} = 0$, solve for V_m:

$$V_m = \frac{g_e \bar{g}_e E_e + g_i \bar{g}_i E_i + g_l \bar{g}_l E_l}{g_e \bar{g}_e + g_i \bar{g}_i + g_l \bar{g}_l}$$

Can now solve for the equilibrium potential as a function of inputs.

Membrane potential computes a *balance* (weighted average) of excitatory and inhibitory inputs.

Equilibrium Potential Illustrated



Computational Neurons (Units)

1. Really abstract: The standard sigmoidal function.
2. More neuro: The point neuron function.
3. Two kinds of outputs: discrete spiking, rate coded.

Abstract Neural Nets

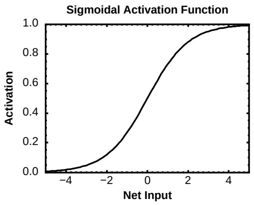
Steps 1 and 3

1. Compute weighted, summed *net input*:

$$\eta_j = \sum_i x_i w_{ij}$$

3. Pass through *sigmoidal* function to compute output:

$$y_j = \frac{1}{1+e^{-\eta_j}}$$



Captures upper and lower limits to activation.
Misses anything tied to V_m (e.g., shunting inhibition).

Bio Neural Nets

Three steps:

1. Compute weighted, summed *net input*:

$$\eta_j \approx \sum_i x_i w_{ij} \approx g_e$$

2. Compute V_m:

$$V_m = \frac{g_e \bar{g}_e E_e + g_i \bar{g}_i E_i + g_l \bar{g}_l E_l}{g_e \bar{g}_e + g_i \bar{g}_i + g_l \bar{g}_l}$$

3. Compute output as: Spikes, or rate code equiv.
Or, rate code via *sigmoidal* function:

$$y_j = \frac{1}{1+(\gamma[V_m(t)-\Theta]_+)^{-1}}$$

Computing V_m

Use $V_m(t+1) = V_m(t) + dt_{vm} I_{net-}$ with biological or normalized (0-1) parameters:

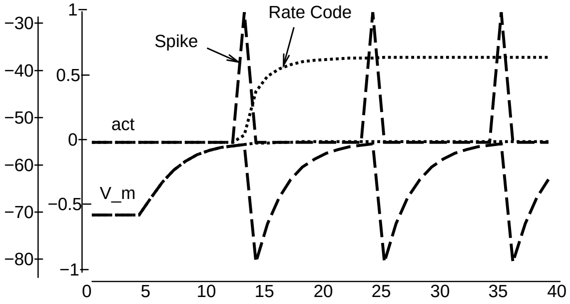
Parameter	mV	(0-1)
V_{rest}	-70	0.15
$E_l\ (K^+)$	-70	0.15
$E_i\ (Cl^-)$	-70	0.15
Θ	-55	0.25
$E_e\ (Na^+)$	+55	1.00

Normalized used by default.

Thresholded Spike Outputs

Voltage gated Na^+ channels open if $V_m > \Theta$, sharp rise in V_m.

Voltage Gated K^+ channels open to reset spike.



In model: $y_j = 1$ if $V_m > \Theta$, then reset (also keep track of rate).

Rate Coded Output

Output *is* average firing rate value.
One unit = % spikes in population of neurons?

Rate approximated by X-over-X-plus-1 ($\frac{x}{x+1}$):

$$y_j = \frac{\gamma[V_m(t)-\Theta]_+}{\gamma[V_m(t)-\Theta]_+ + 1}$$

which is like a sigmoidal function:

$$y_j = \frac{1}{1+(\gamma[V_m(t)-\Theta]_+)^{-1}}$$

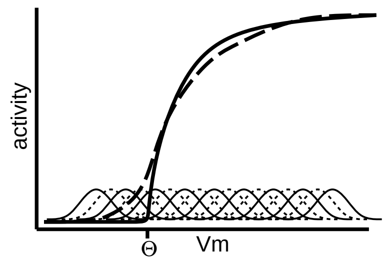
compare to sigmoid: $y_j = \frac{1}{1+e^{-\eta_j}}$

γ is the *gain*: makes things sharper or duller.

Convolution with Noise

X-over-X-plus-1 has a very sharp threshold

Smooth by *convolve* with noise (just like “blurring” or “smoothing” in an image manip program):



Fit of Rate Code to Spikes