NN & FL

Introduction

Artificial Intelligence

- Artificial Intelligence (AI) is a branch of science that is concerned with the automation of intelligent behavior.

- it is possible to build machines that can demonstrate intelligence similar to human beings.

- A I can be obtained in two ways.
  - soft computing methods (NN, FL, GA etc)
  - Hard computing methods (conventional PID cont. etc)

Artificial Intelligence

- Hard computing methods are predominantly based on mathematical approaches .
- Soft computing techniques have drawn their inherent characteristics from biological systems
- Soft computing methods are
  - Neural networks
  - Fuzzy logic
  - Genetic algorithms
  - Combination of above

Neural Networks

- Neural Networks (NN) are simplified models of the biological nervous systems
- An NN can be massively parallel and therefore is said to exhibit parallel distributed processing

- NN architectures have been broadly classified as
  - Single layer feed forward networks.
  - Multi layer feed forward networks and

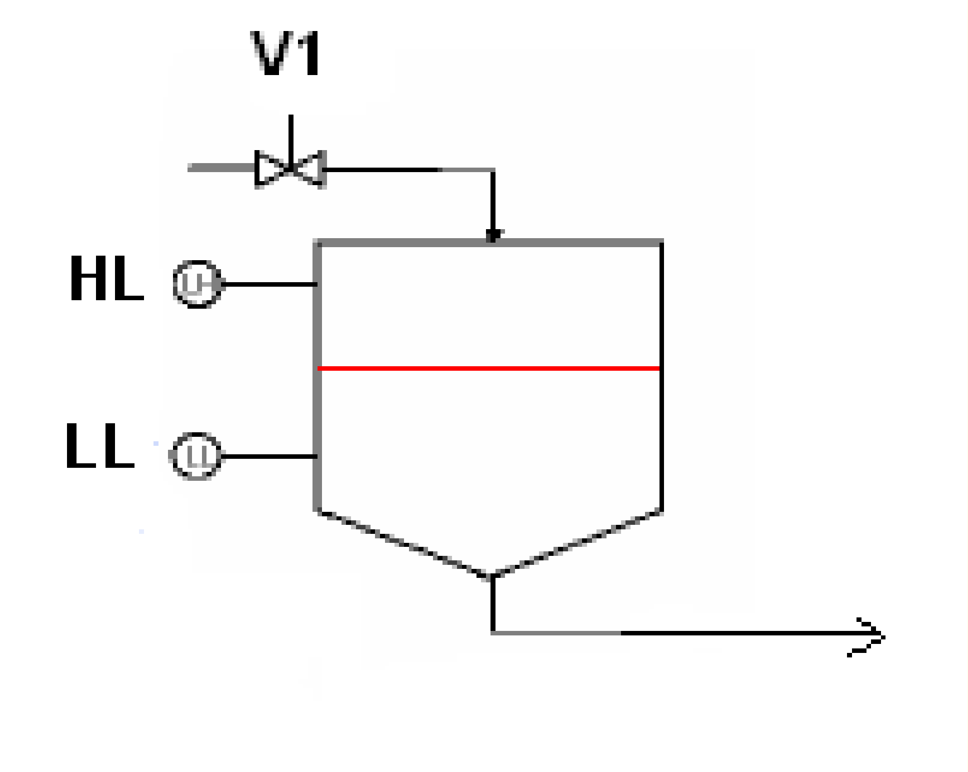
Fuzzy Logic

- Fuzzy logic is a set of mathematical principles for knowledge representation based on the membership function

- Fuzzy logic provides simple way to draw definite conclusions from vague, ambiguous or imprecise information.

- Fuzzy logic is similar to that of Boolean logic

Fuzzy Logic

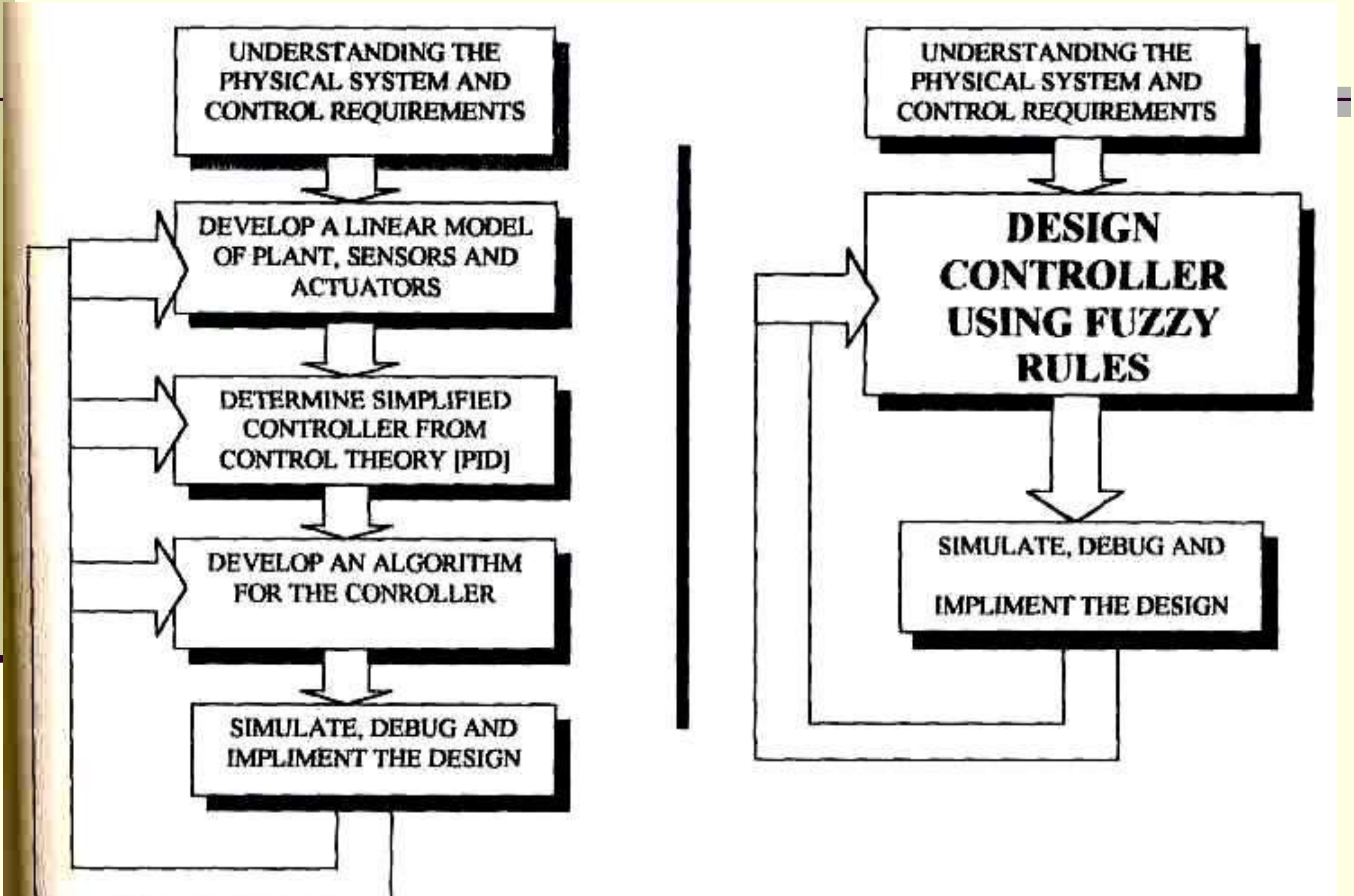


If the Level is ***low*** then open V1
If the Level is ***High*** then Close V1
If the level is ***Medium*** then open V1 by HALF (50%)

Need of fuzzy logic controller

- Rigorous mathematical model of some linear process.
- in the case of complex process, which are difficult to model.
- Non-Linear Systems

Comparison of conventional & fuzzy logic controllers:



Fuzzy Logic

Sets & Fuzzy Sets

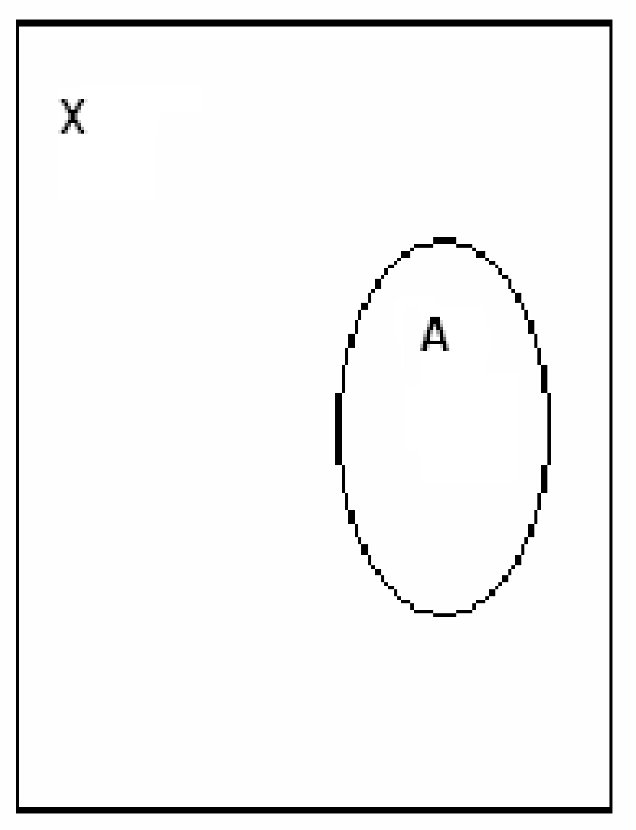
Set (Crisp set)

- Well defined collection of objects

- If **X** is Universe of discourse (Universal set)

A is any set from **X**.

x is any element in **X**



Set & membership function

Def: Let X be the universe of discourse and its elements be denoted as x.

In the classical set theory, crisp set **A** of X is defined as: $f_A(x)$ Called membership function of **A**

$$f_A(x) : X \rightarrow \{0,1\}$$

where $$f_A(x) = \begin{cases} 1, & if \cdot x \in A \\ 0 & if \cdot x \notin A \end{cases}$$

membership function (crisp set)

- Crisp set of “ tall persons”

Degree of
Membership

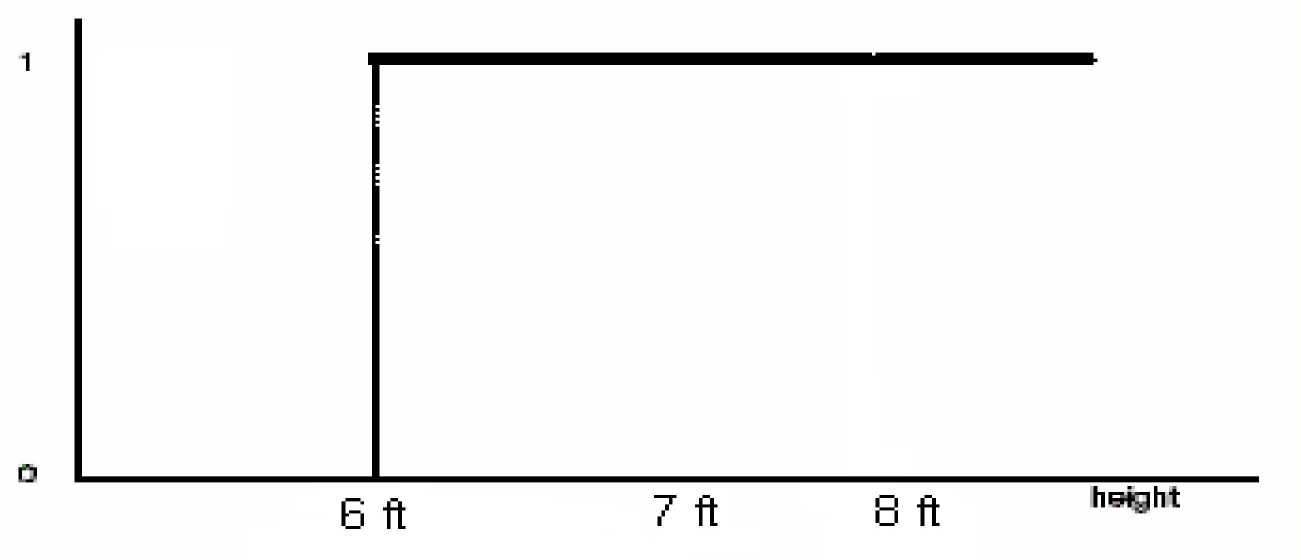


Figure 1: A crisp way of modeling tallness

membership function (crisp set)

Crisp set of " Short persons" or " NOT tall "

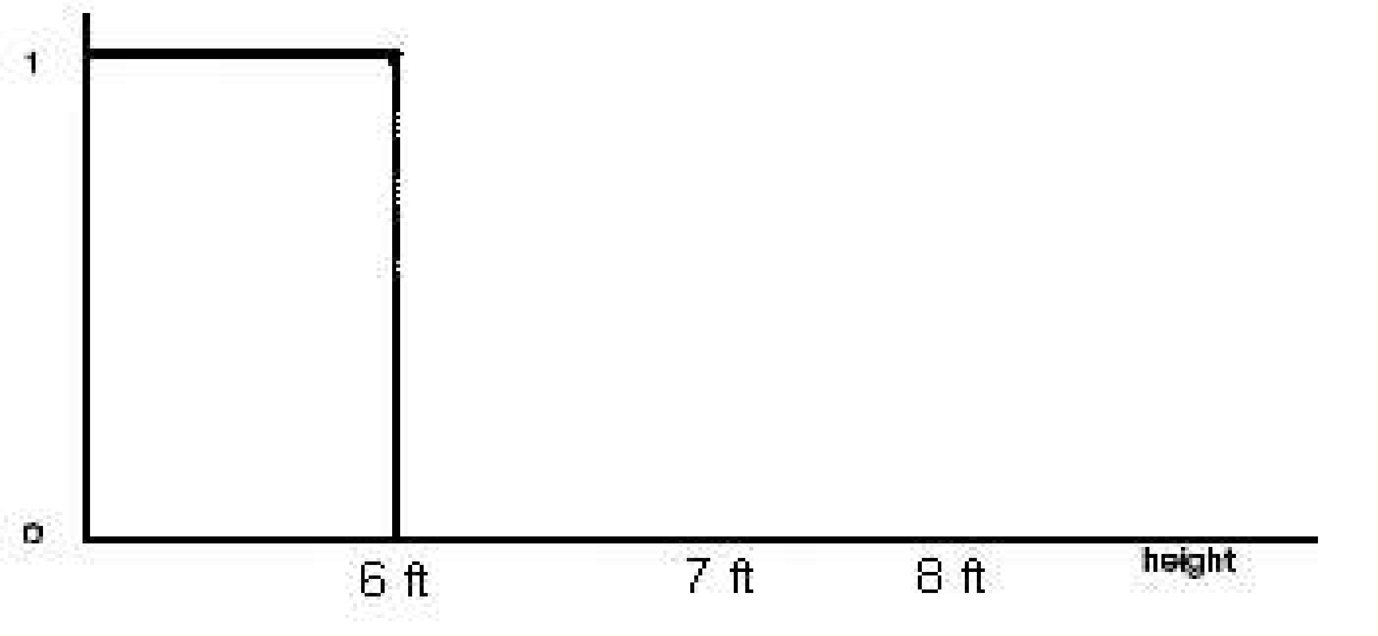
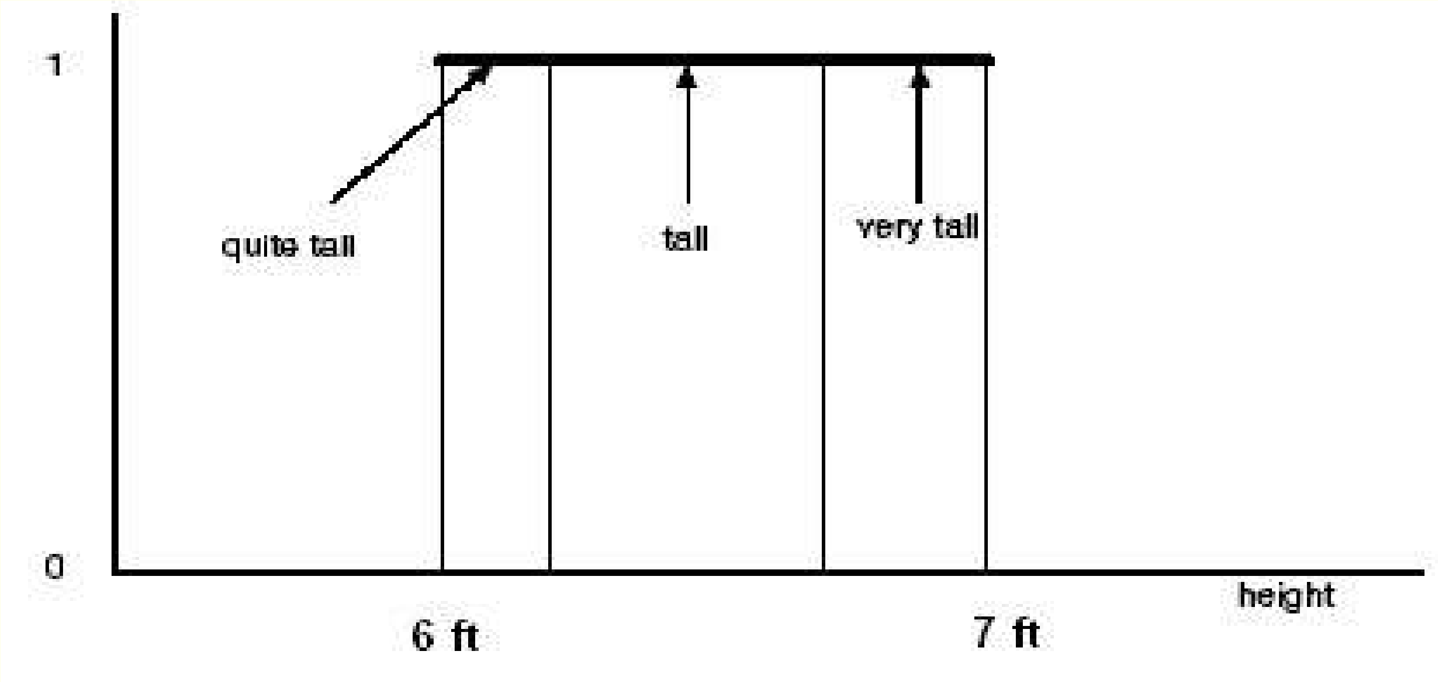


Figure 2: The crisp version of short

membership function (crisp set)

- Different heights have same ‘tallness’



Fuzzy Set

Def: Let X be the universe of discourse and its elements be denoted as x.

In the fuzzy set theory, fuzzy set **A** of X is defined as: $\mu_A(x)$ Called membership function of fuzzy set **A**

$$\mu_A(x): X \rightarrow [0, \quad 1]$$

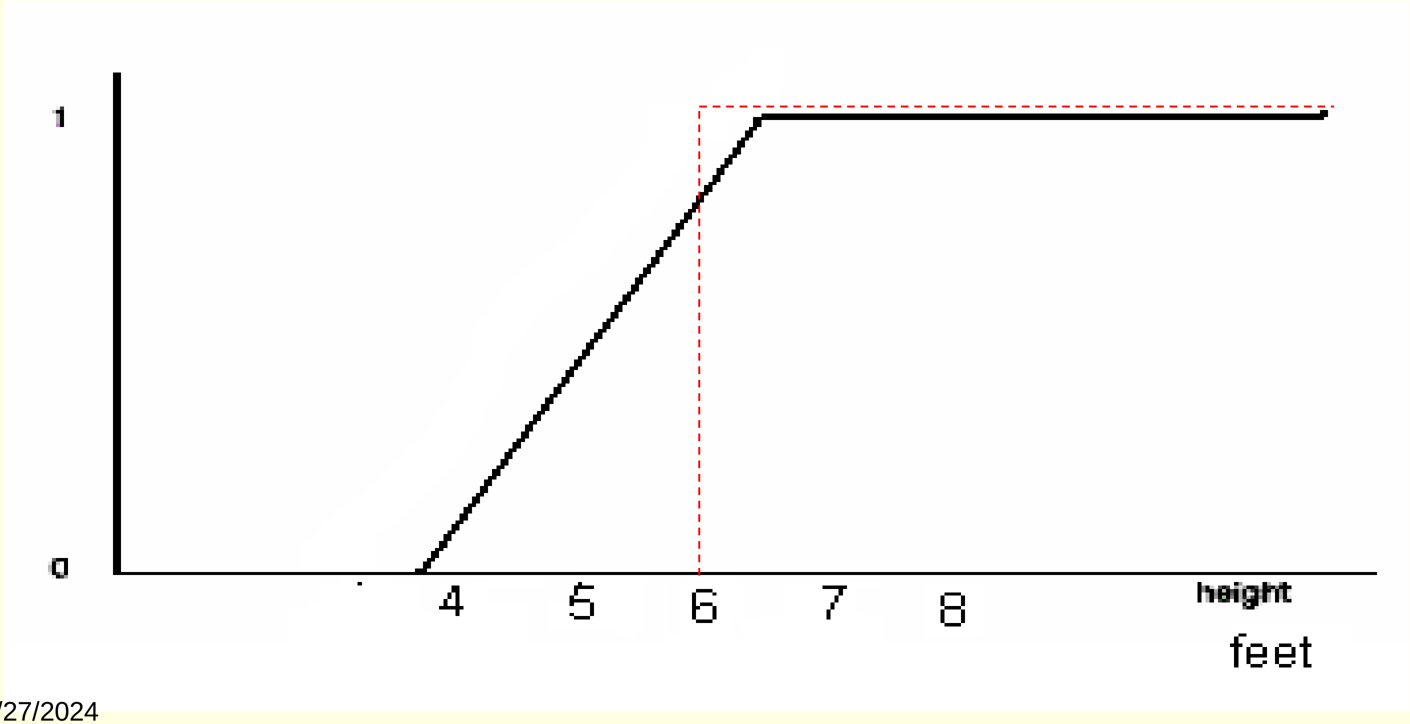
Where

$$\mu_A(x) = \begin{cases} 1, & \text{if } x \text{ is totally in } A \\ 0, & \text{if } x \text{ is not in } A \\ 0 \prec \mu_A(x) \prec 1, & \text{if } x \text{ is partially in } A \end{cases}$$

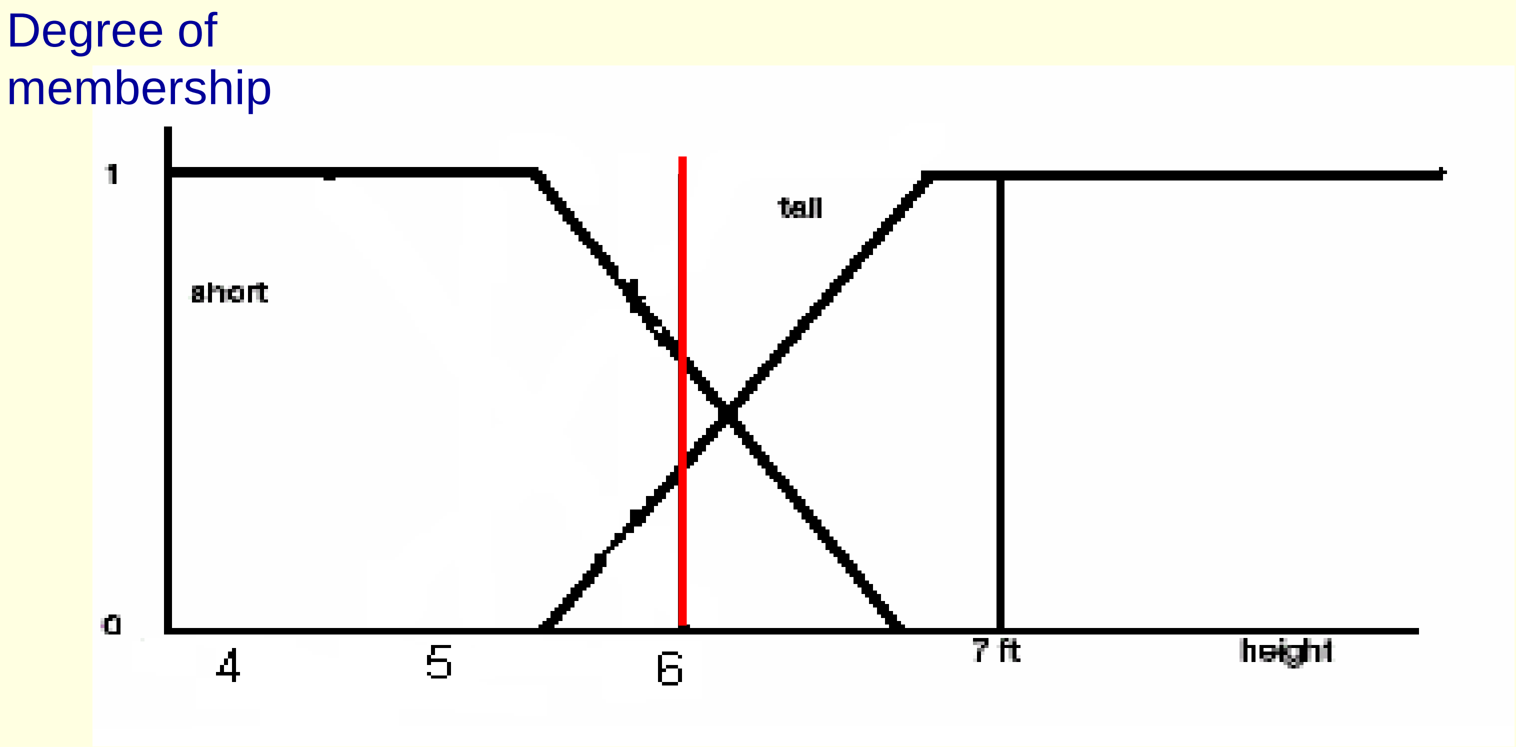
Fuzzy Sets & membership

- The shape you see is known as the membership function

Degree of
membership



Fuzzy Sets & memberships



Shows two membership functions: ‘tall’ and ‘short’

Notation

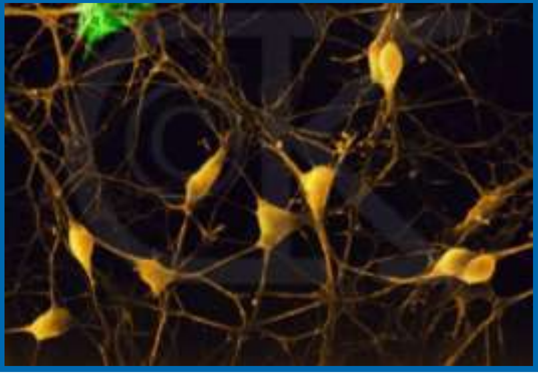
Crisp Set $$A = \{\, x / P(x) \,\}$$

Fuzzy Set $$A = \{\, (\, x, \mu_A(x) \,) \,/ x \in X \,\}$$

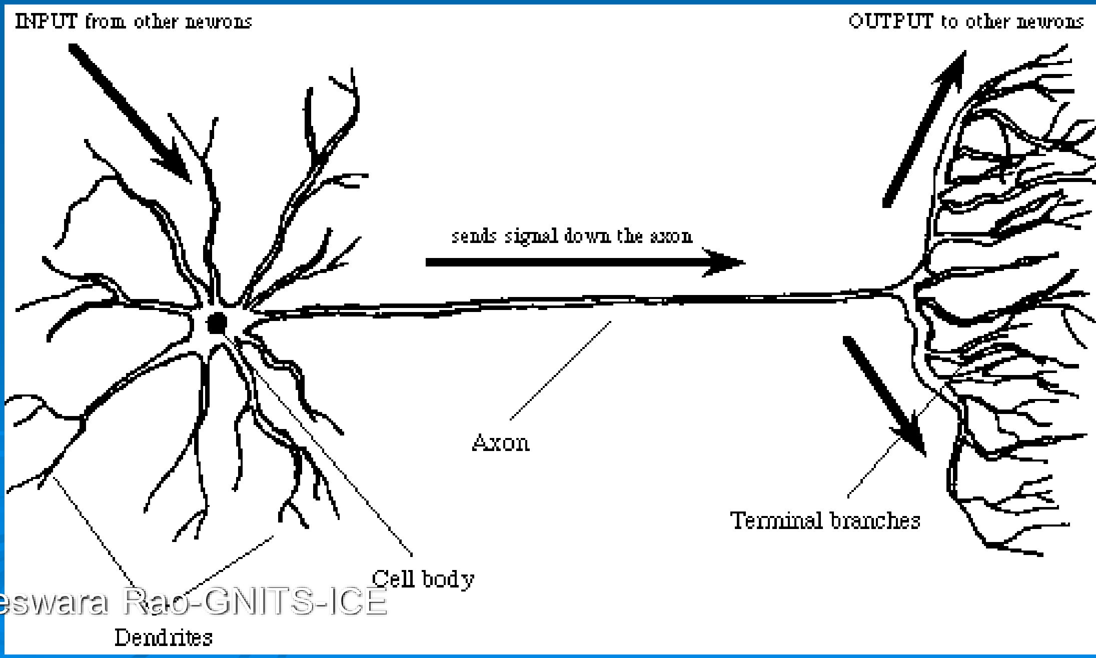
Biological Neuron
&
Neuron Models

The Biological Neuron

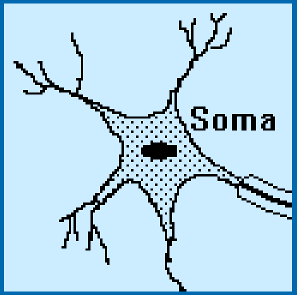
- Human Nervous system → 1.3x10^{10} neurons
  - 10^{10} are in brain
  - Each connected to ~10,000 other neurons
  - Power dissipation ~20W



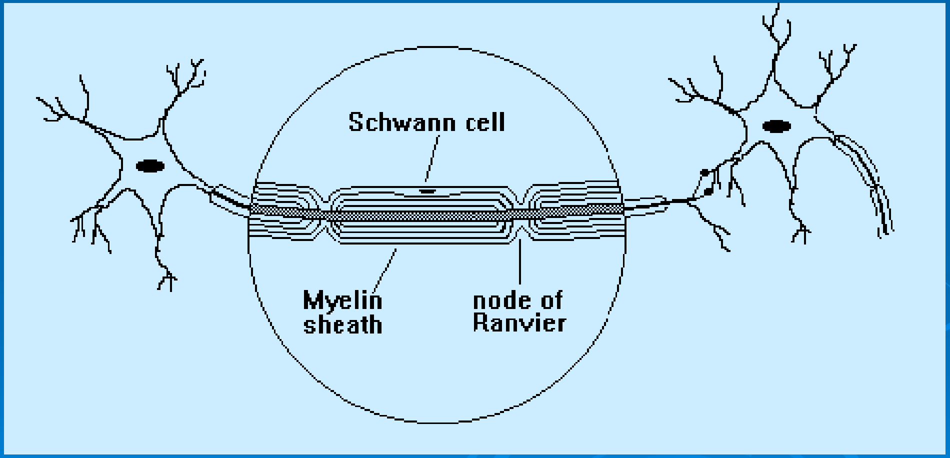
- Neuron Structure:
  - Cell Body – Soma
  - Axon/Nerve Fibers
  - Dendrites
  - Presynaptic Terminals



The Biological Neuron

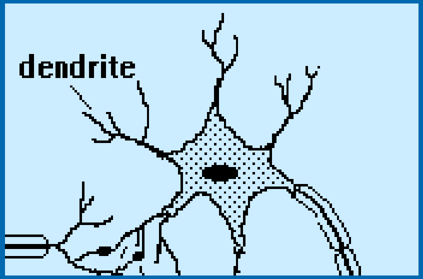


- Cell Body – Soma
  - Includes Nucleus & Perikaryon
  - Metabolic Functions
  - Generates the transmission signal (*action potential)* – through *axon hillock -*, when received signal threshold reached
- Axon/Nerve Fibers
  - Conduction Component
  - 1 per neuron
  - 1mm to 1m
  - Extends from *axon hillock* to *terminal buttons*
  - Smooth surface
  - No ribosome

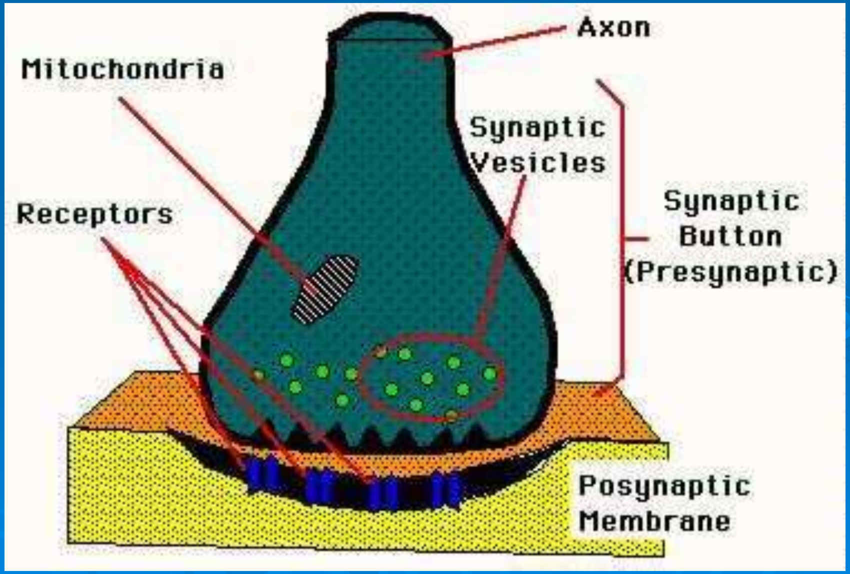


The Biological Neuron

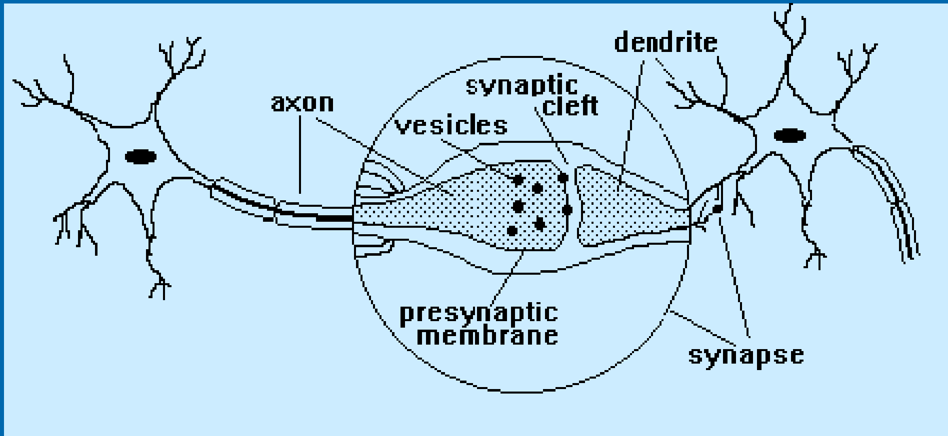
- **Dendrites**
  - The receiver / input ports
  - Several Branched
  - Rough Surface (dendritic spines)
  - Have ribosomes
  - No myelin insulation



- **Presynaptic Terminals**
  - The branched ends of axons
  - Transmit the signal to other neurons' dendrites with *neurotransmitters*



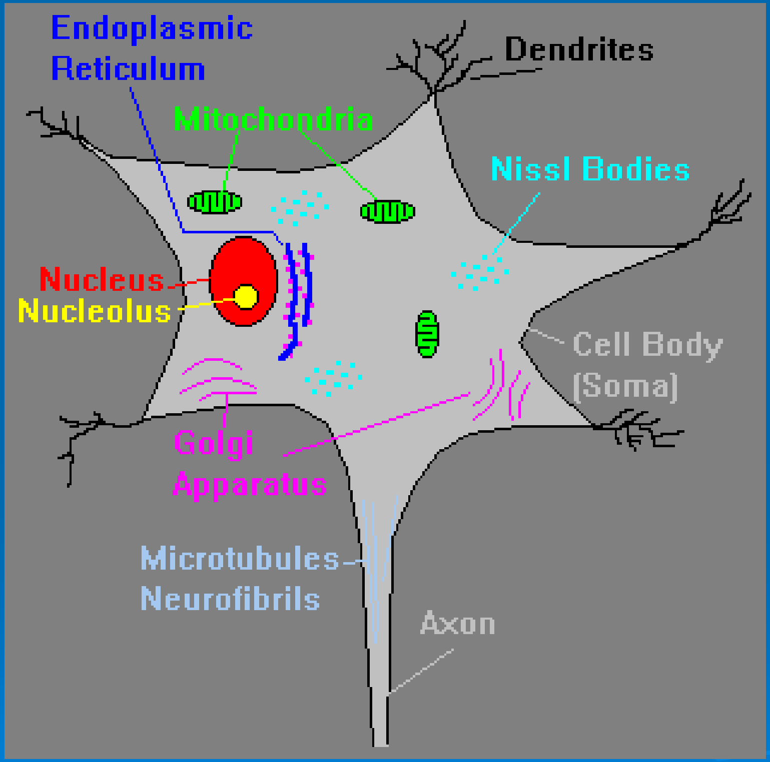
The Biological Neuron



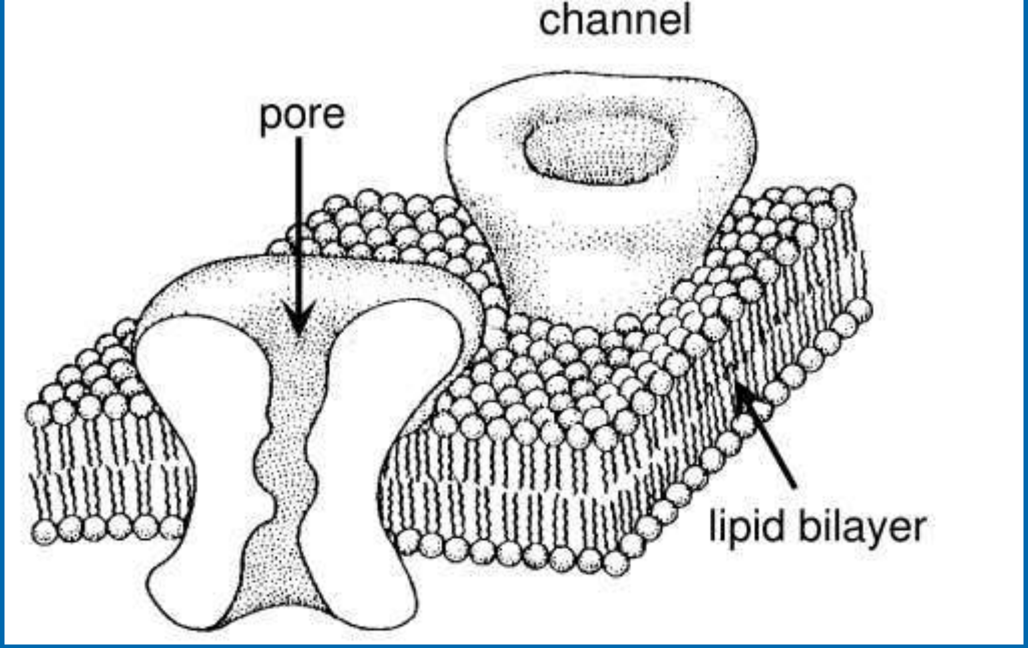
- **Synapse:**
  - Junction of 2 neurons
  - Signal communication
  - Two ways of transmission:
    - Coupling of ion channels → Electrical Synapse
    - Release of chemical transmitters → Chemical Synapse
- Chemical Synapse:
  - Presynaptic neuron releases *neurotransmitters* through synaptic vesicles at terminal button to the synaptic cleft – the gap between two neurons.
  - Dendrite receives the signal via its receptors
  - [Excitatory & Inhibitory Synapses – Later]

The Biological Neuron

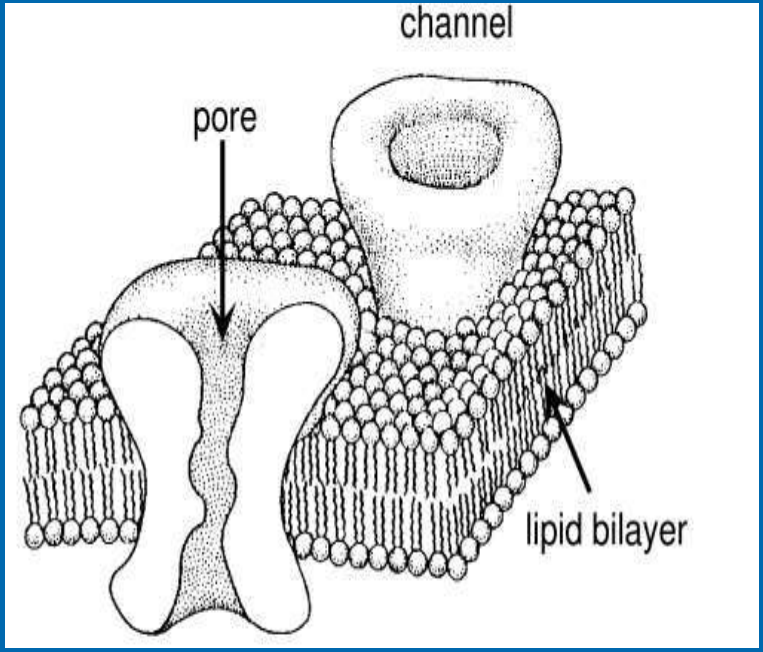
- Inside of a Neuron:

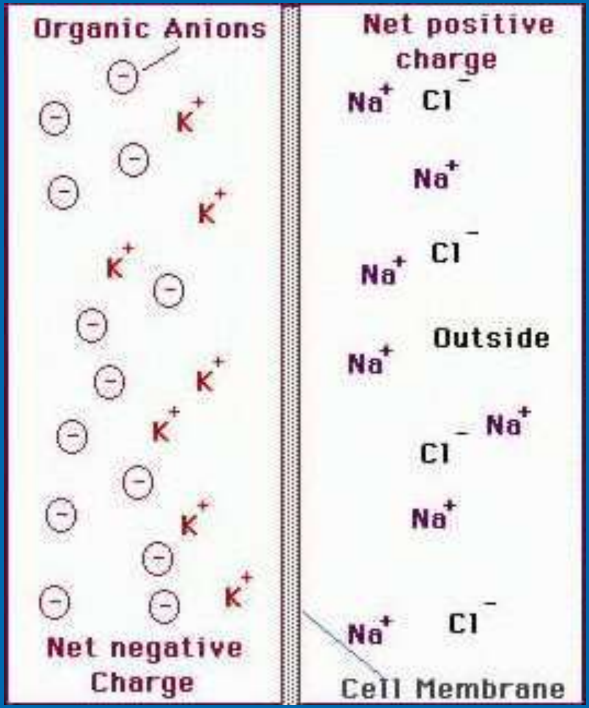


- **Nucleus - genetic material (chromosomes)**
- **Endoplasmic reticulum (ER) - system of tubes → material transport in cytoplasm**
- **Golgi Apparatus - membrane-bound structure → packaging peptides and proteins (including neurotransmitters) into vesicles**
- **Microfilaments/Neurotubules - transport for materials within neuron & structural support.**
- **Mitochondria - Produce energy**



- Neurons are enclosed by a membrane separating interior from extra cellular space
- The concentration of ions inside is different (more –ve) to that in the surrounding liquid





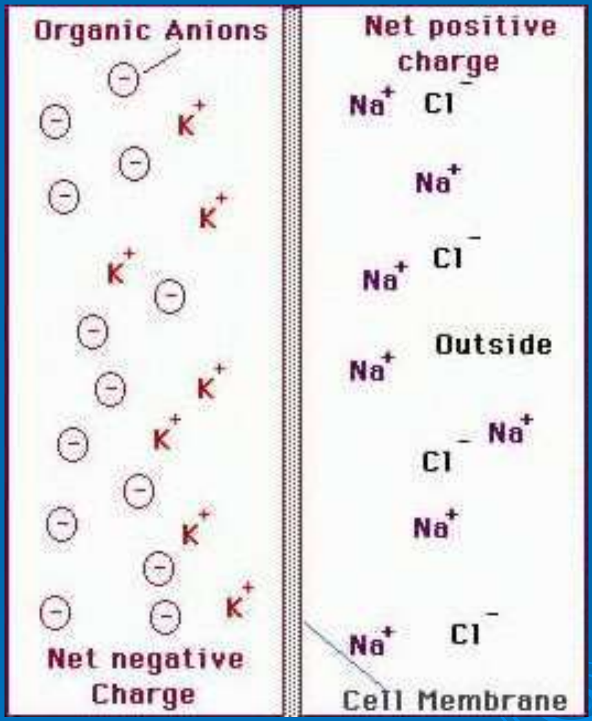
- -ve ions therefore build up on the inside surface of the membrane and an equal amount of +ve ions build up on the outside
- The difference in concentration generates an electrical potential (membrane potential) which plays an important role in neuronal dynamics.

The Biological Neuron

- **Membrane Potential:**
  - 5nm thick, semipermeable
  - Lipid bilayer controls ion diffusion
  - Potential difference ~70 mV
  - Charge pump:
    - Na^+ →
    - ← K^+

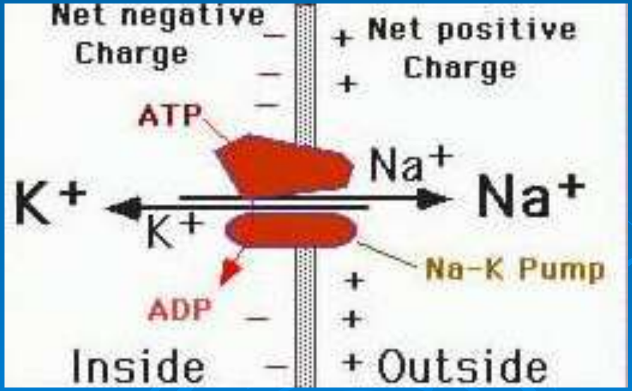
→: Outside Cell
←: Into Cell

- **Resting Potential:**
  - When no signaling activity
  - Outside potential defined 0
    - ➔ Vr = ~ -70mV



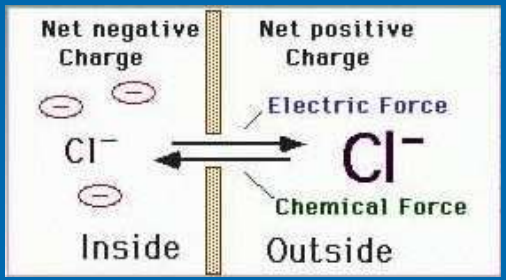
The Biological Neuron

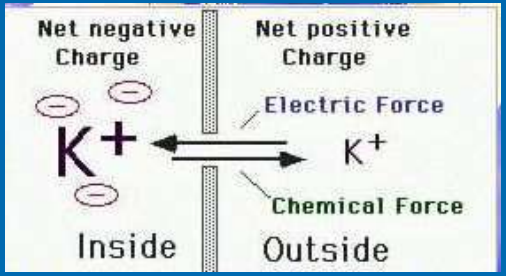
- Membrane Potential – Charge Distribution:
  - Inside: More K^+ & Organic Anions (acids & proteins)
  - Outside: More Na^+ & Cl^-
  - 4 Mechanisms that maintain charge distribution = membrane potential:
    - 1) Ion Channels:
      Ion distribution 🡐 channel distribution
    - 2) Chemical Concentration Gradient
      - Move toward low gradient
    - 3) Electrostatic Force
      - Move along/against E-Field
    - 4) Na-K Pumps
      - Move Na & K against their net electrochemical gradients

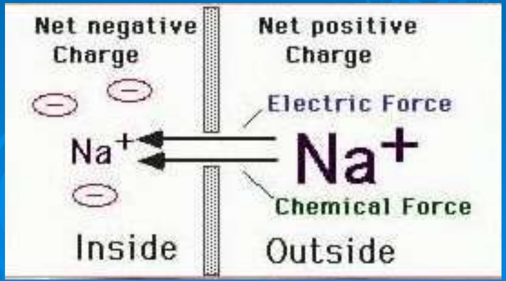


The Biological Neuron

- Membrane Potential – Charge Distribution:
  - Cl^- :
    - Concentration gradient ←
    - Electrostatic Force →
    - Final concentration depends on membrane potential
  - K^+ :
    - Concentration gradient →
    - Electrostatic Force ←
    - Na-K pump ←
  - Na^+ :
    - Concentration gradient ←
    - Electrostatic Force ←
    - Na-K pump →





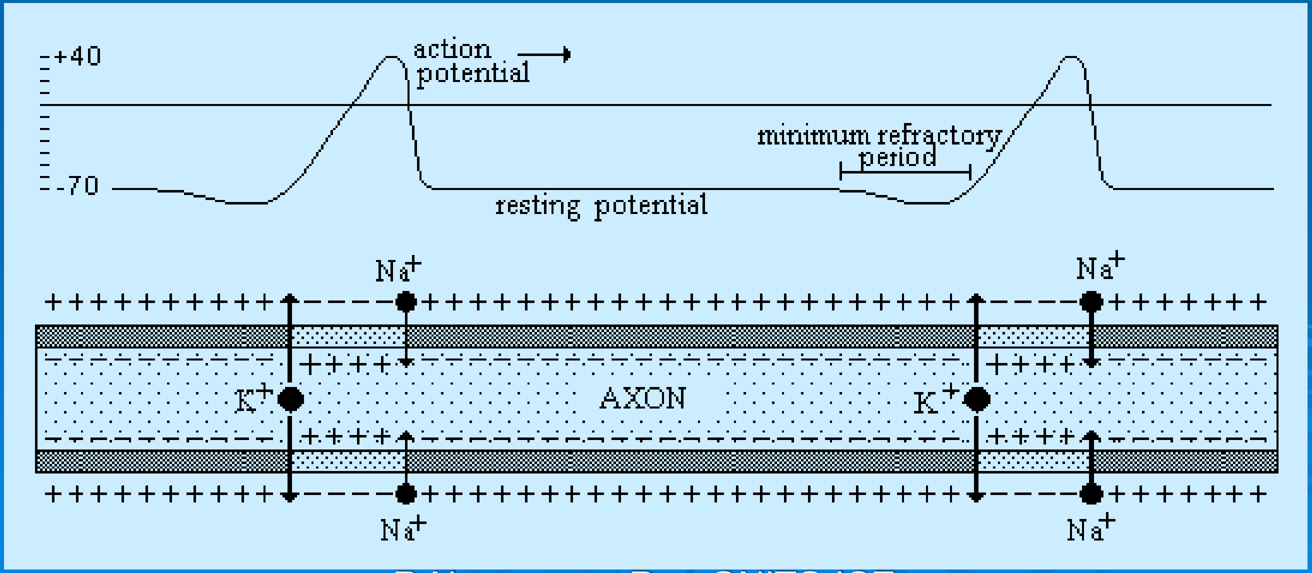


The Biological Neuron

- Excitatory & Inhibitory Synapses:
  - Neurotransmitters → Receptor sites at postsynaptic membrane
  - Neurotransmitter types
    - Increase Na-K pump efficiency
      - → Hyperpolarization
    - Decrease Na-K pump efficiency
      - → Depolarization
  - Excitatory Synapse:
    - Encourage depolarization
      ← Activation decreases Na-K pump efficiency
  - Inhibitory Synapse:
    - Encourage hyperpolarization
      ← Activation increases Na-K pump efficiency

The Biological Neuron

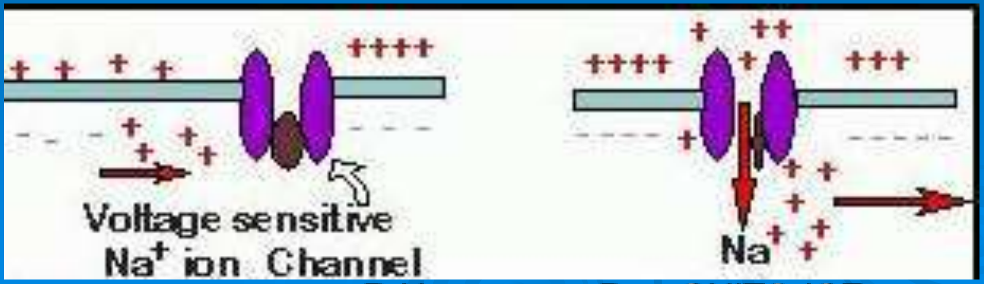
- **Action Potential:**
  - Short reversal in membrane potential
    - → Current flow: Action Potential → Rest Potential
    - → Propagation of the depolarization along axon



The Biological Neuron

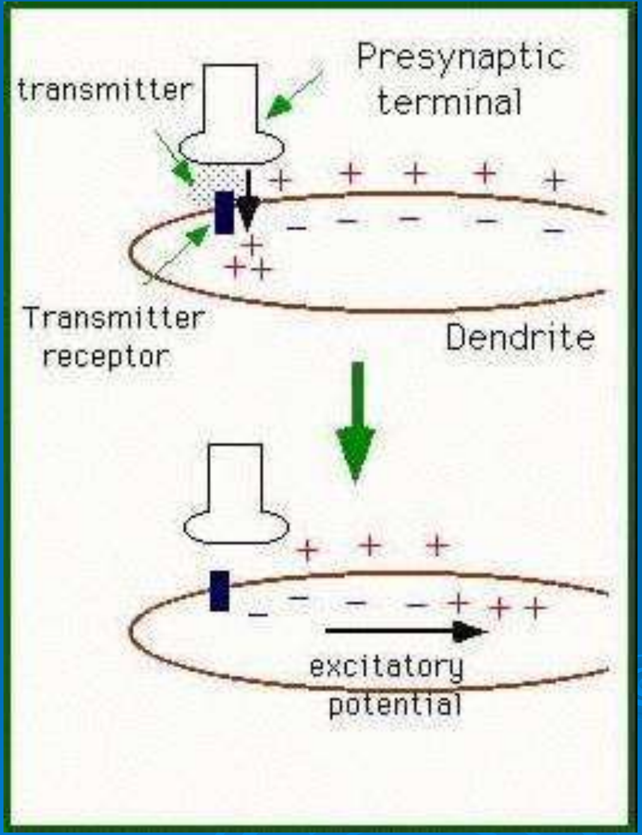
- Action Potential:
  - Sufficient Excitatory Synapses Activation – Depolarization of Soma
    → trigger action potential:
    - Some Voltage gated Na Channels open
      → Membrane Na Permeability Increases
      → ← Na^+ → Depolarization increases
    - Depolarization builds up exponentially…

Positive Feedback



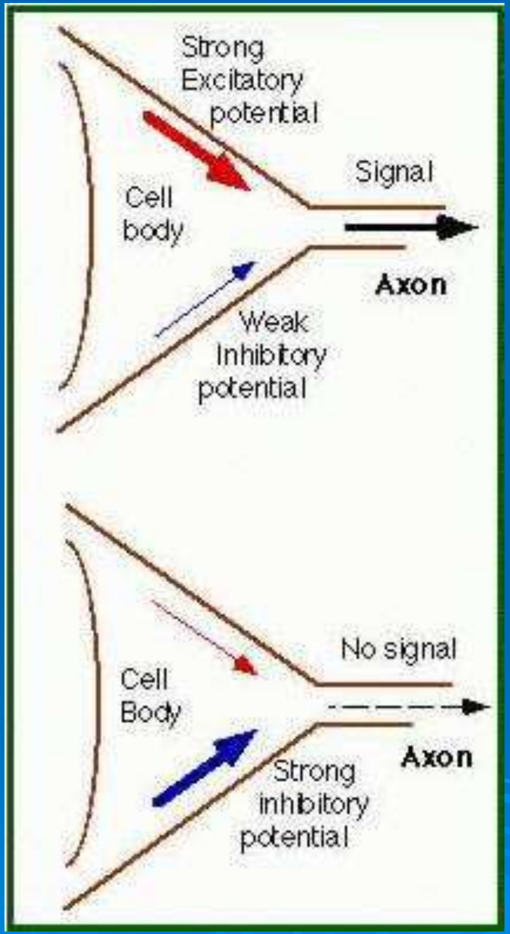
Biological Neuron: Processing of Signals

- A cell at rest maintains an electrical potential difference known as the resting potential with respect to the outside.
- An incoming signal perturbs the potential inside the cell. Excitatory signals depolarizes the cell by allowing positive charge to rush in, inhibitory signals cause hyper-polarization by the in-rush of negative charge.



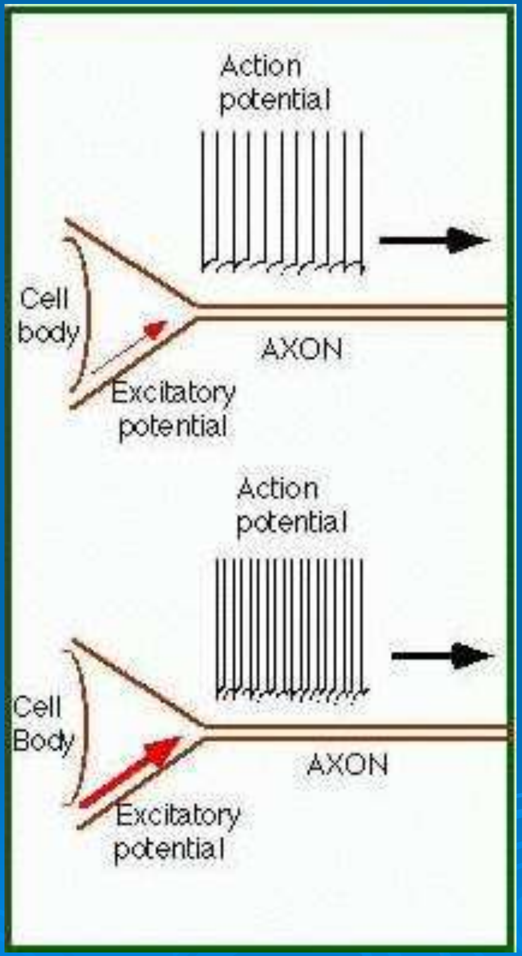
Biological Neuron: Processing of Signals

- Voltage sensitive sodium channels trigger possibly multiple “action potentials” or voltage spikes with amplitude of about 110mV depending on the input.



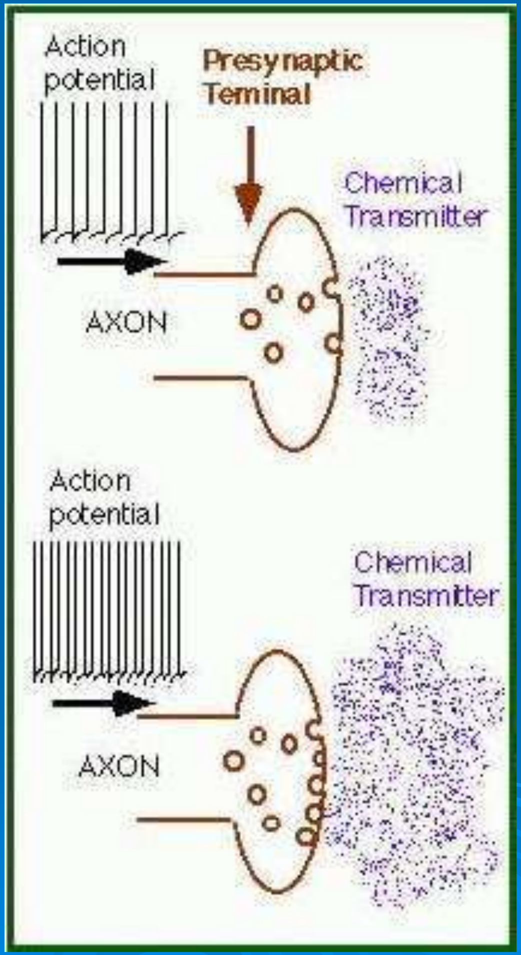
Biological Neuron: Conduction in Axon

- Axon transmits the action potential, regenerating the signal to prevent signal degradation.
- Conduction speed ranges from 1m/s to 100m/s. Axons with myelin sheaths around them conduct signals faster.
- Axons can be as long as 1 meter.



Biological Neuron: Output of Signal

- At the end of the axon, chemicals known as neurotransmitters are released when excited by action potentials.
- Amount released is a function of the frequency of the action potentials. Type of neurotransmitter released varies by type of neuron.



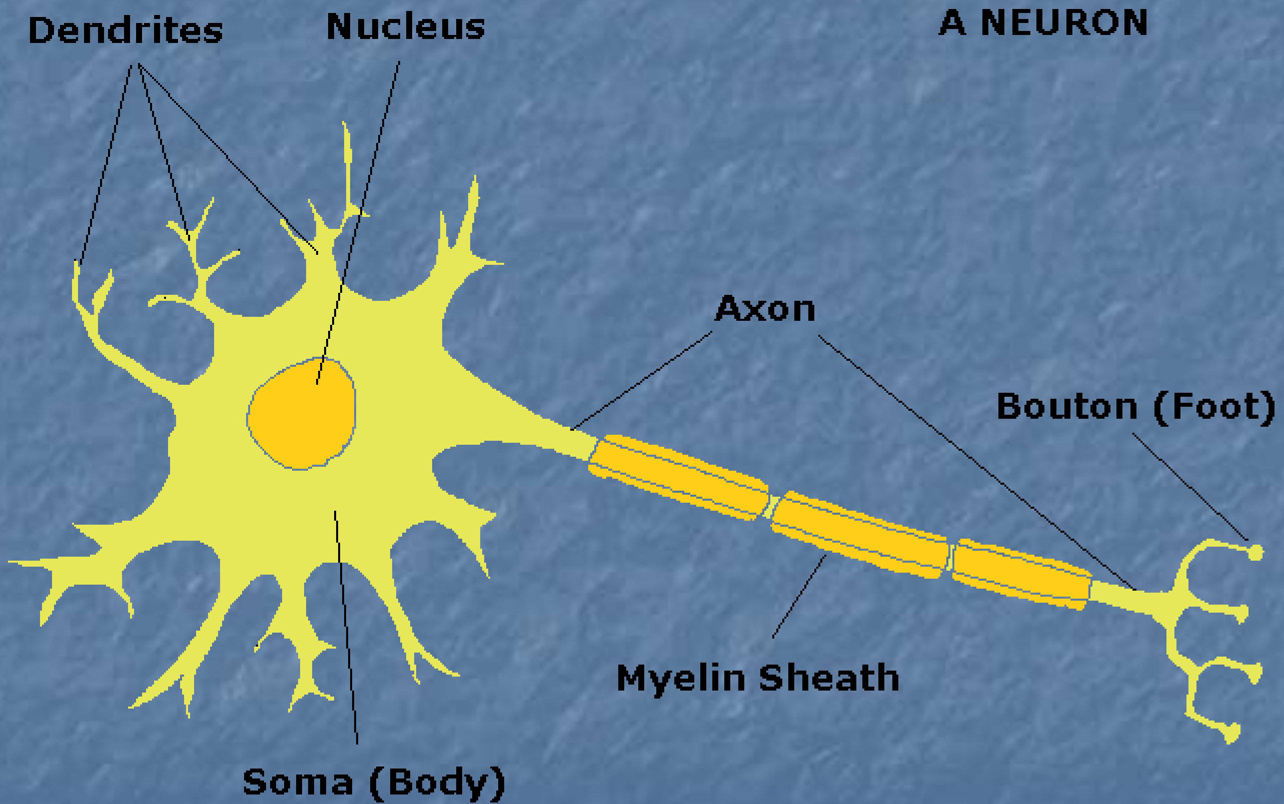
Hodgkin-Huxley Model

Nervous System

- Signals are propagated from nerve cell to nerve cell (*neuron*) via electro-chemical mechanisms

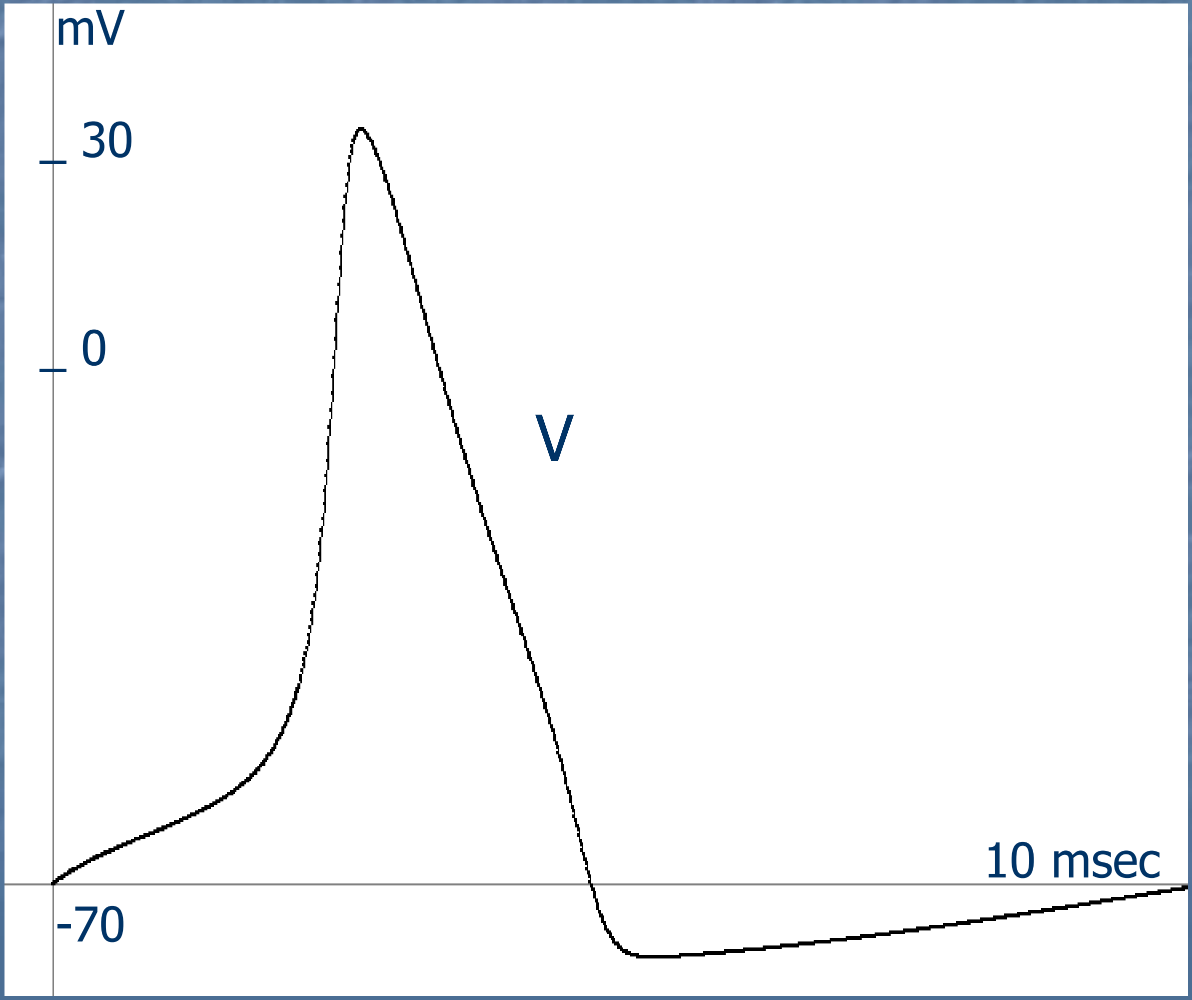
- Hodgkin and Huxley experimented on squids and discovered how the signal is produced within the neuron
- three different types of ion current, viz., sodium, potassium, and a leak current that consists mainly of Cl- ions.

Neuron



C. George Boeree: www.ship.edu/~cgboeree/

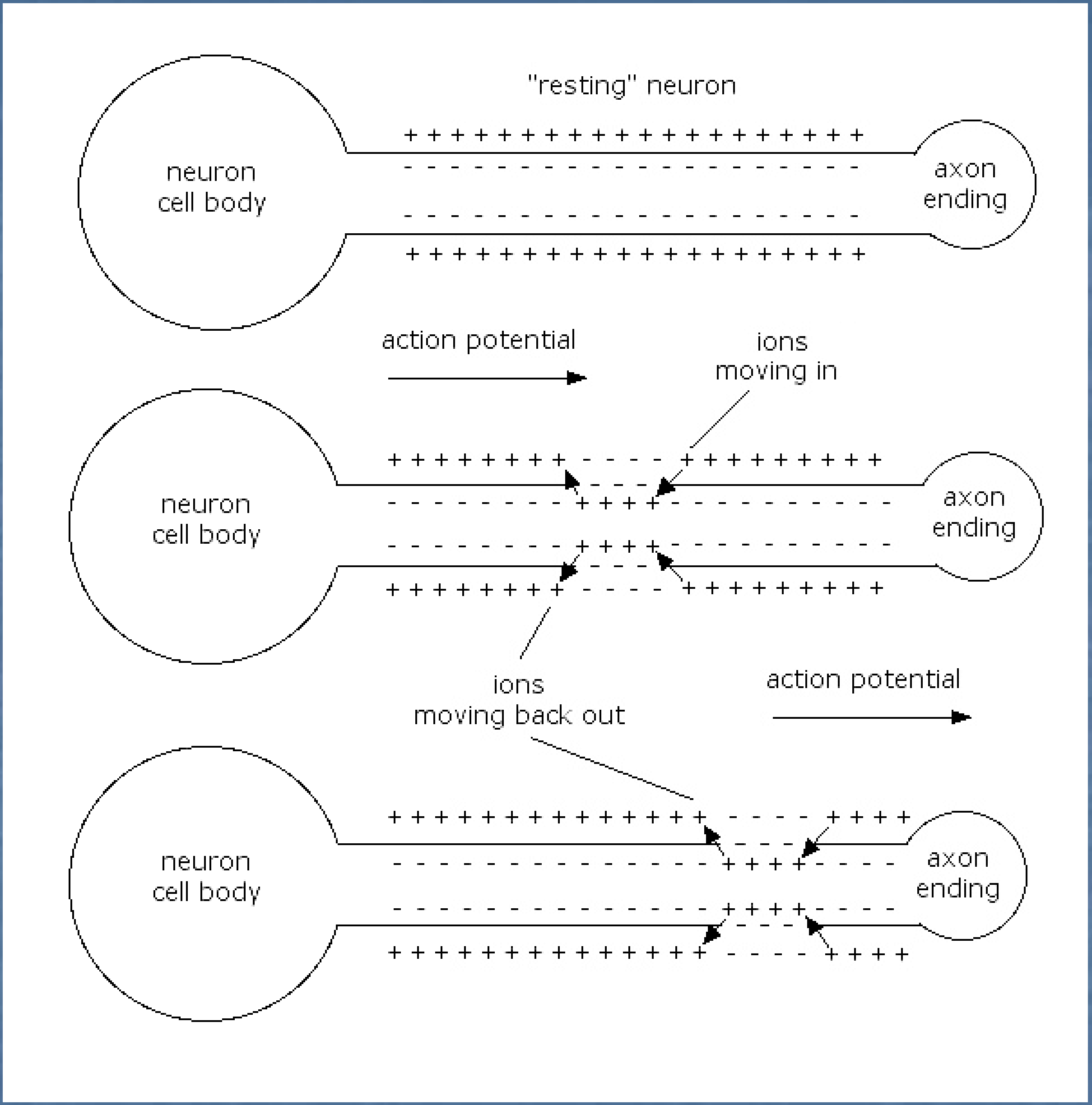
Action Potential



Axon membrane potential difference

$$V = V_i - V_e$$

When the axon is excited, V spikes because sodium Na+ and potassium K+ ions flow through the membrane.



Nernst Potential

V_{Na} , V_K and V_r

Ion flow due to electrical signal

Traveling wave

C. George Boeree: www.ship.edu/~cgboeree/

Hodgkin-Huxley Model

- The semipermeable cell membrane separates the interior of the cell from the extracellular liquid and acts as a capacitor.
- If an input current $I(t)$ is injected into the cell, it may add further charge on the capacitor, or leak through the channels in the cell membrane.
- Because of active ion transport through the cell membrane, the ion concentration inside the cell is different from that in the extracellular liquid.

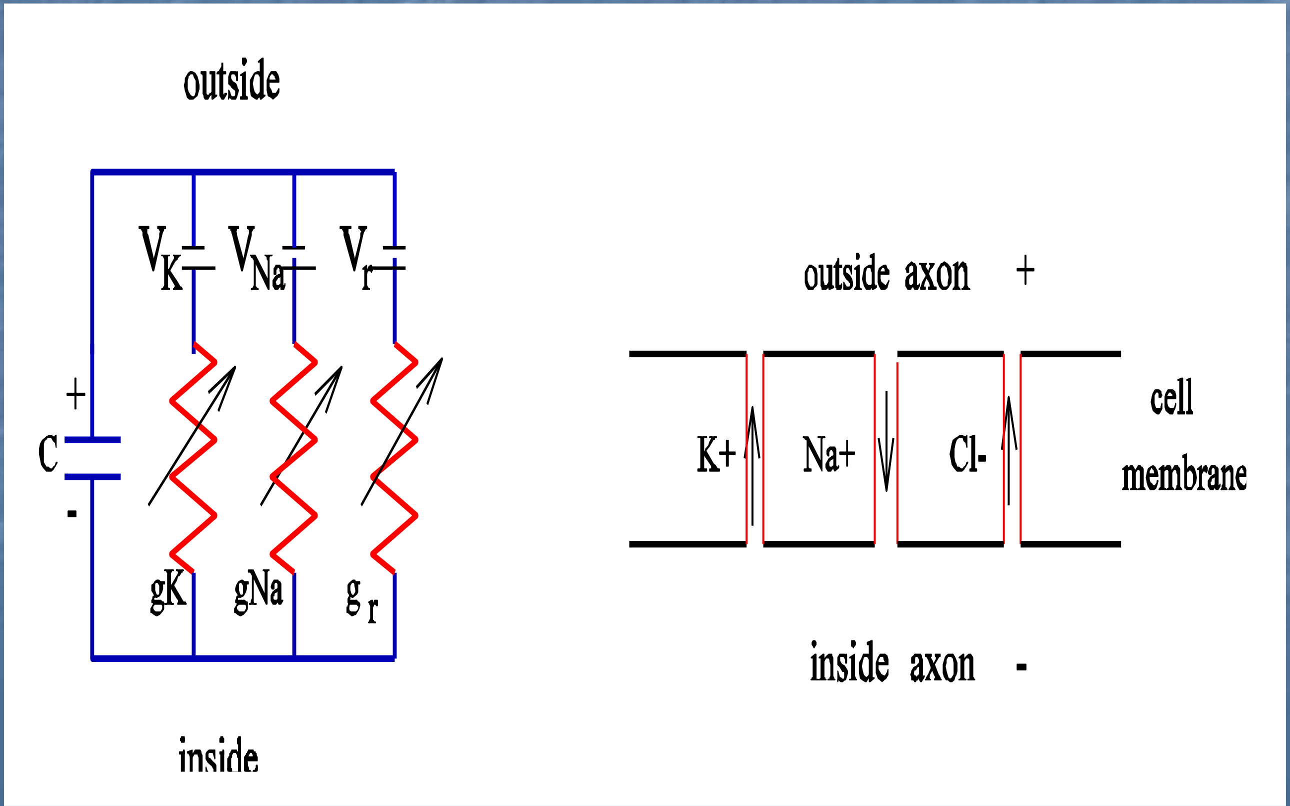
Hodgkin-Huxley Model

- the applied current $I(t)$ may be split in a capacitive current I_C which charges the capacitor C and further components I_k which pass through the ion channels.

Circuit Model for Axon Membrane

Since the membrane separates charge, it is modeled as a capacitor with capacitance C. Ion channels are resistors.

$$1/R = g = \text{conductance}$$



$i_C = C\, dV/dt$

$i_{Na} = g_{Na}\,(V - V_{Na})$

$i_K = g_K\,(V - V_K)$

$i_r = g_r\,(V - V_r)$

Circuit Equations

Since the sum of the currents is 0, it follows that

$$C\frac{dV}{dt} = -g_{Na}(V - V_{Na}) - g_K(V - V_K) - g_r(V - V_r) + I_{ap}$$

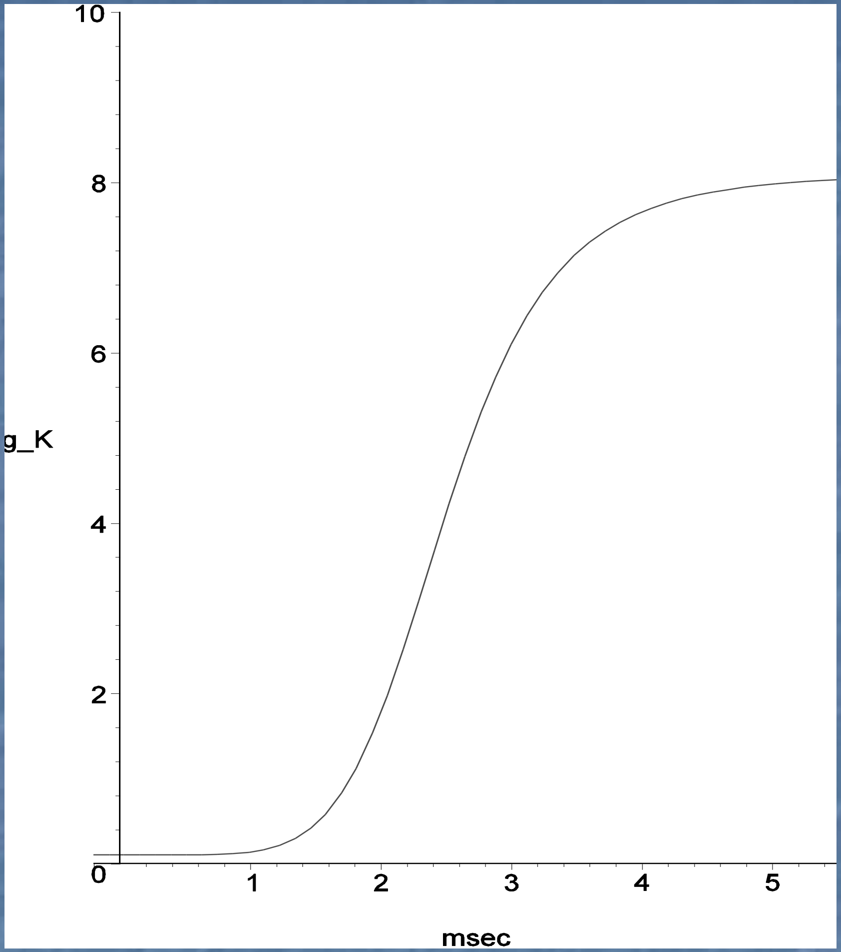
where I_{ap} is applied current. If ion conductances are constants then group constants to obtain 1[st] order, linear eq

$$C\frac{dV}{dt} = -g(V - V^*) + I_{ap}$$

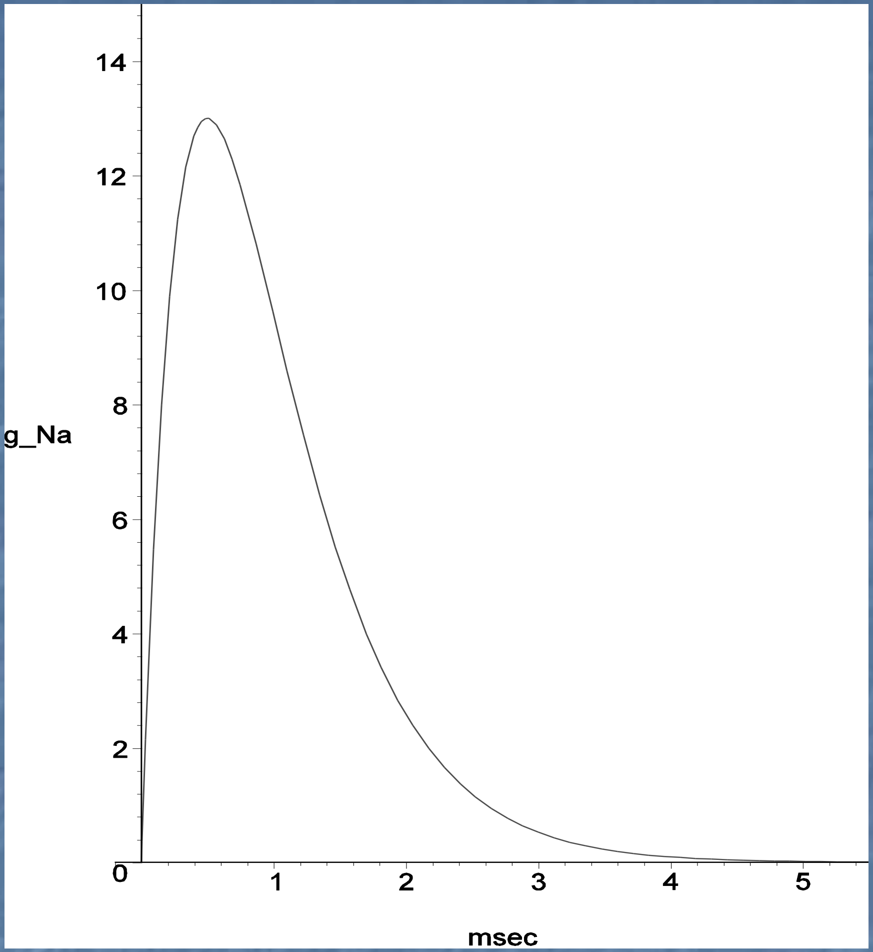
Solving gives

$$V(t) \rightarrow V^* + I_{ap}/g$$

Variable Conductance



g



Experiments showed that g_{Na} and g_K varied with time and V. After stimulus, Na responds much more rapidly than K .

Hodgkin-Huxley System

Four state variables are used:

$v(t)=V(t)-V_{eq}$ is membrane potential,

$m(t)$ is Na activation,

$n(t)$ is K activation and

$h(t)$ is Na inactivation.

In terms of these variables $g_K=\underline{g}_K n^4$ and $g_{Na}=\underline{g}_{Na} m^3 h$. The resting potential $V_{eq} \approx -70$mV. Voltage clamp experiments determined g_K and n as functions of t and hence the parameter dependences on v in the differential eq. for $n(t)$. Likewise for $m(t)$ and $h(t)$.

Hodgkin-Huxley System

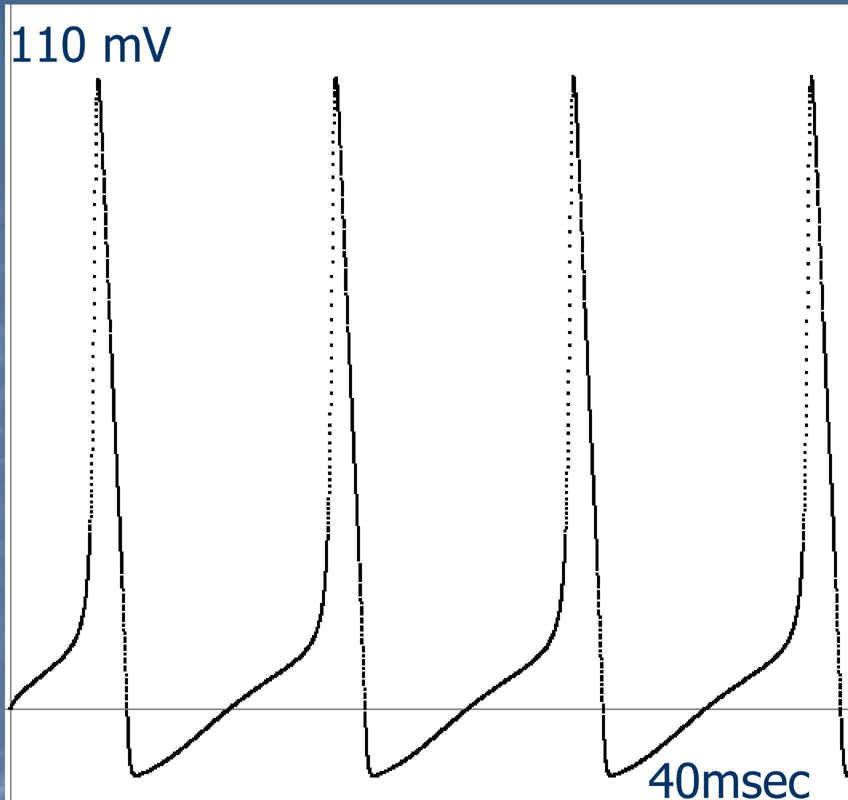
$$C\frac{dv}{dt} = -\underline{g}_{Na} m^3 h(v - V_{Na}) - \underline{g}_{K} n^4 (v - V_K) - g_r(v - V_r) + I_{ap}$$

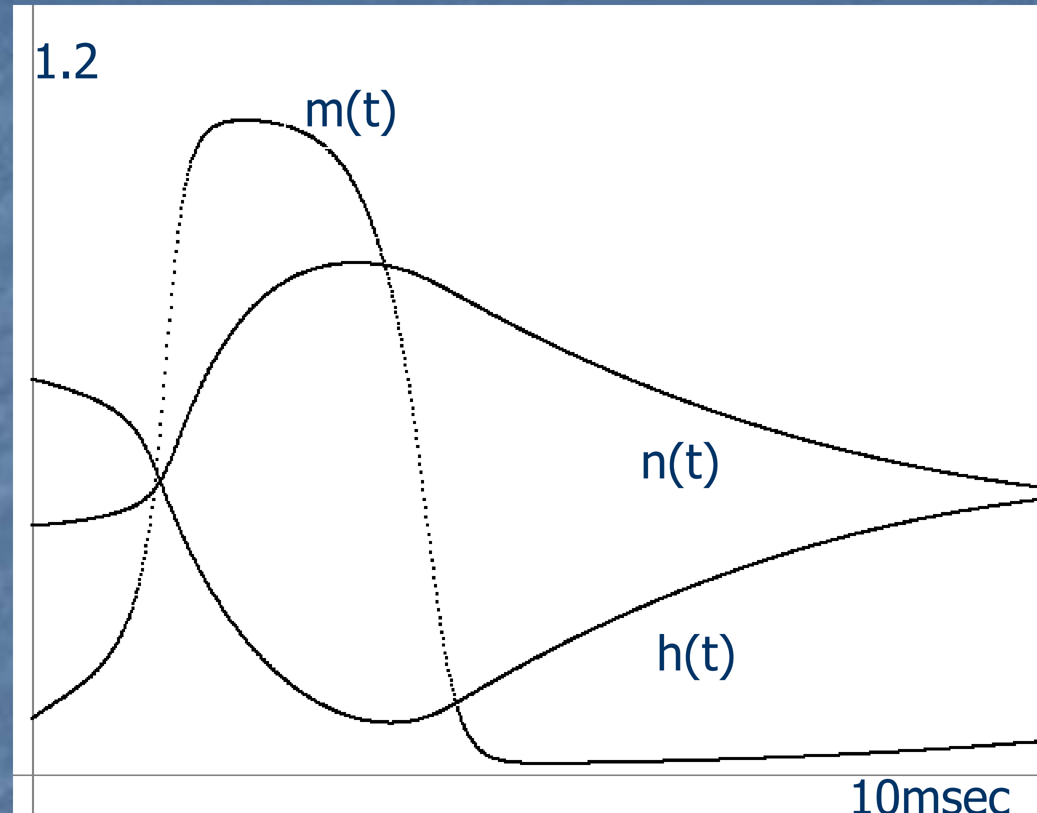
$$\frac{dm}{dt} = \alpha_m(v)(1 - m) - \beta_m(v)m$$

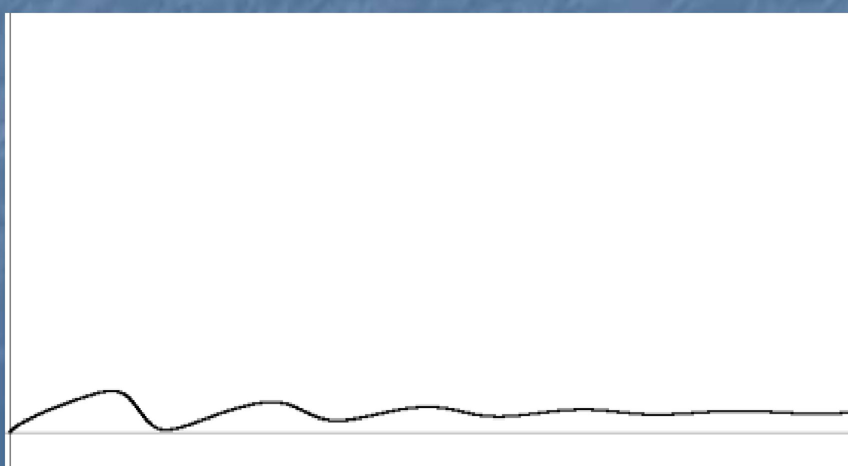
$$\frac{dn}{dt} = \alpha_n(v)(1 - n) - \beta_n(v)n$$

$$\frac{dh}{dt} = \alpha_h(v)(1 - h) - \beta_h(v)h$$

I_{ap} =8, v(t)

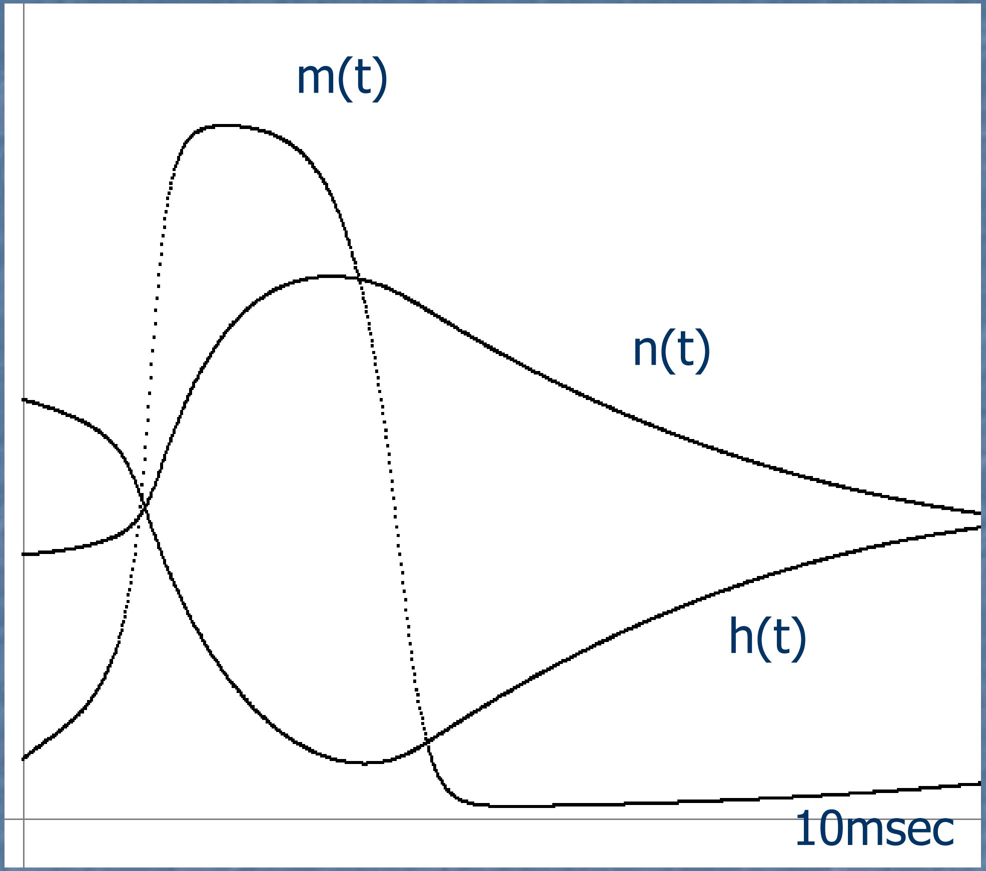






I_{ap}=7, v(t)

Fast-Slow Dynamics



$\rho_m(v)\ dm/dt = m_\infty(v) - m.$

$\rho_m(v)$ is much smaller than

$\rho_n(v)$ and $\rho_h(v)$. An increase in v results in an increase in $m_\infty(v)$ and a large dm/dt. Hence Na activates more rapidly than K in response to a change in v.

v, m are on a fast time scale and n, h are slow.

FitzHugh-Nagumo System

$$\varepsilon \frac{dv}{dt} = f(v) - w + I \quad \text{and} \quad \frac{dw}{dt} = v - 0.5w$$

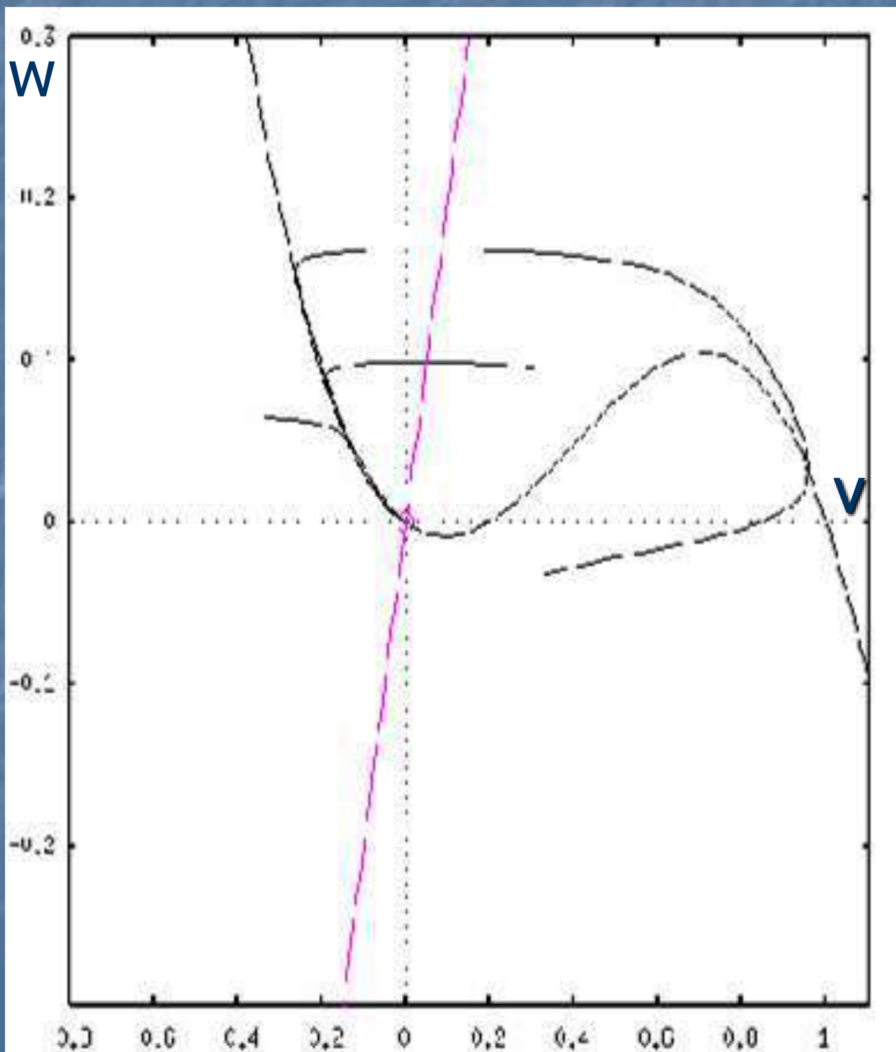
I represents applied current, ε is small and f(v) is a cubic nonlinearity. Observe that in the (v,w) phase plane

$$\frac{dw}{dv} = \frac{\varepsilon(v - 0.5w)}{f(v) - w + I}$$

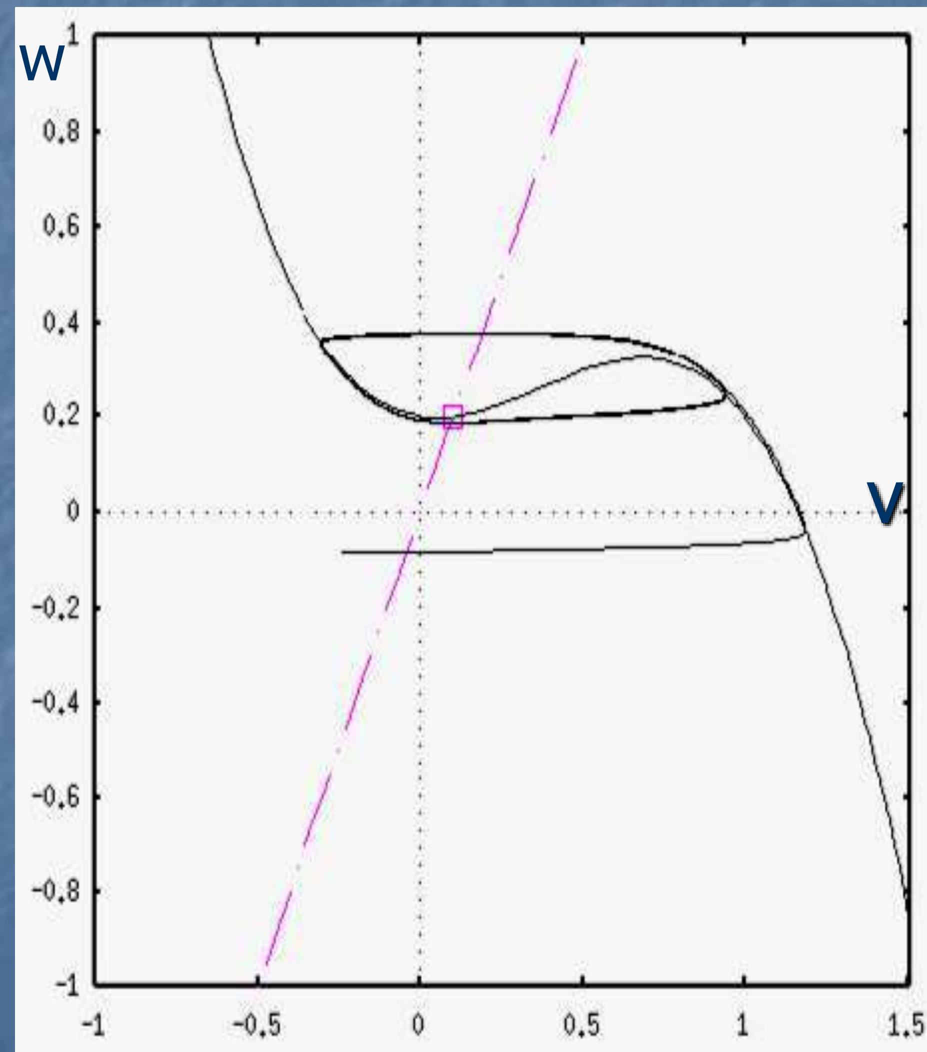
which is small unless the solution is near f(v)-w+*I*=0. Thus the *slow manifold* is the cubic w=f(v)+*I* which is the *nullcline* of the fast variable v. And w is the slow variable with *nullcline* w=2v.

Take f(v)=v(1-v)(v-a) .

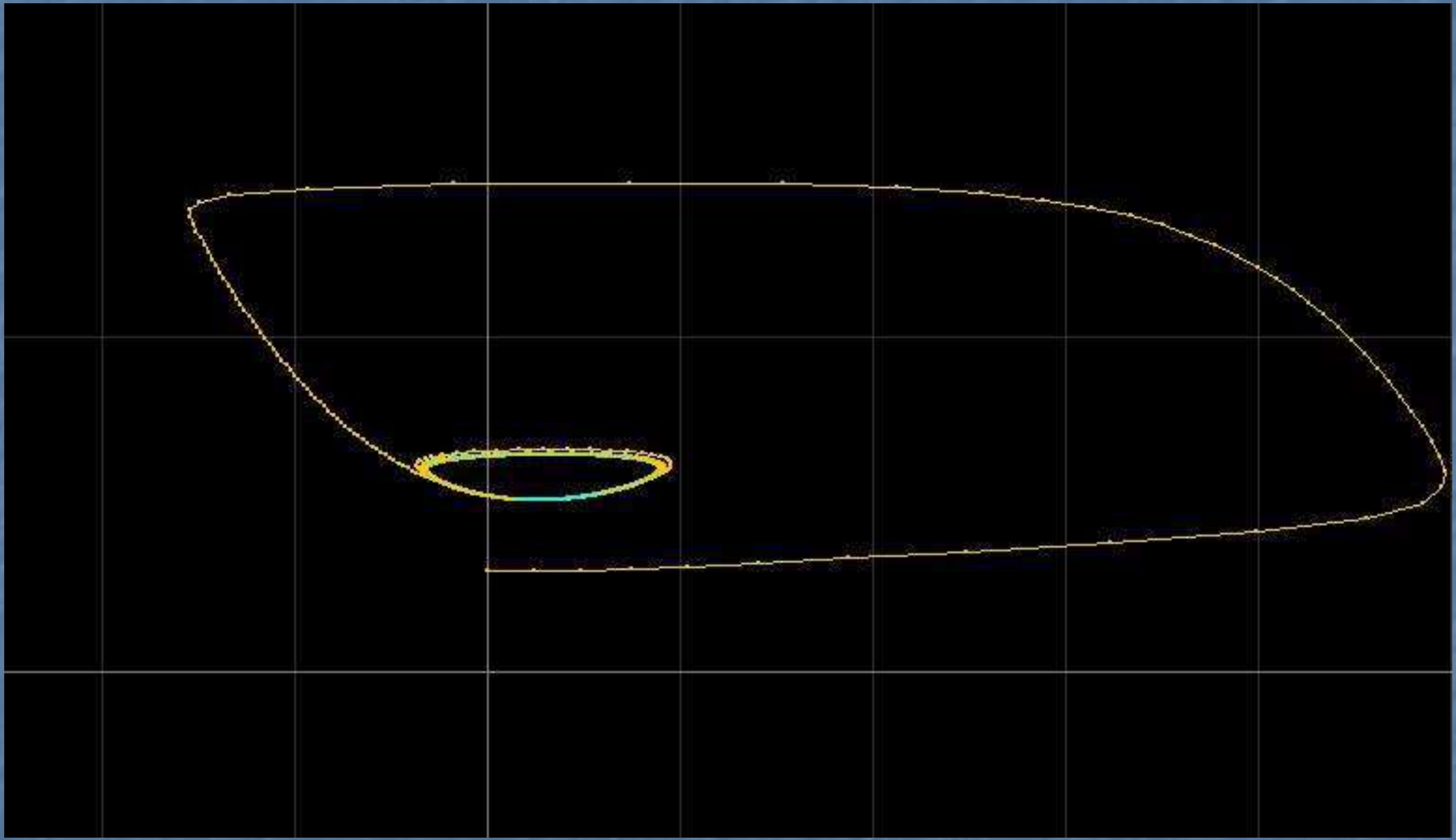
Stable rest state I=0



Stable oscillation I=0.2



FitzHugh-Nagumo Orbits



Single neuron modelling

Integrate and Fire Model

Integrate and fire models

- These models basically assume that action potentials are simply spikes occurring when the membrane potential reaches a threshold V_{th}
- After firing membrane potential is reset to a $V_{reset} < V_{th}$
- Simplifies the modelling dramatically as we only deal with sub threshold membrane potential dynamics
- Can be modelled at various levels of rigour depending on simplifying assumptions used

Membrane capacitance and resistance

- Start by modelling these neurons with assumption that membrane potential is constant

- Denoting membrane capacitance by C_m and the excess charge on the membrane as Q we have:

$$Q = C_m V \quad \text{and} \quad dQ/dt = C_m dV/dt$$

- Shows how much current needed to change membrane potential at a given rate

Membrane capacitance and resistance

- Membrane also has a resistance: R_m

  Determines size of potential difference caused by input of current: $I_e R_m$
- Both R_m and C_m are dependent on surface area of membrane A.
- Membrane time constant $t_m = R_m C_m$ sets the basic time-scale for changes in the membrane potential (typically between 10 and 100ms)

Membrane current

- The membrane current is total current flowing through all the ion channels
- We represent it by i_m which is current/unit area of membrane
- Amount of current flowing each channel is equl to driving force (difference between equilibrium potential E_i and membrane potential) multiplied by channel *conductance* g_i

Therefore:

$$i_m = g_i(V - E_i)$$

Membrane current

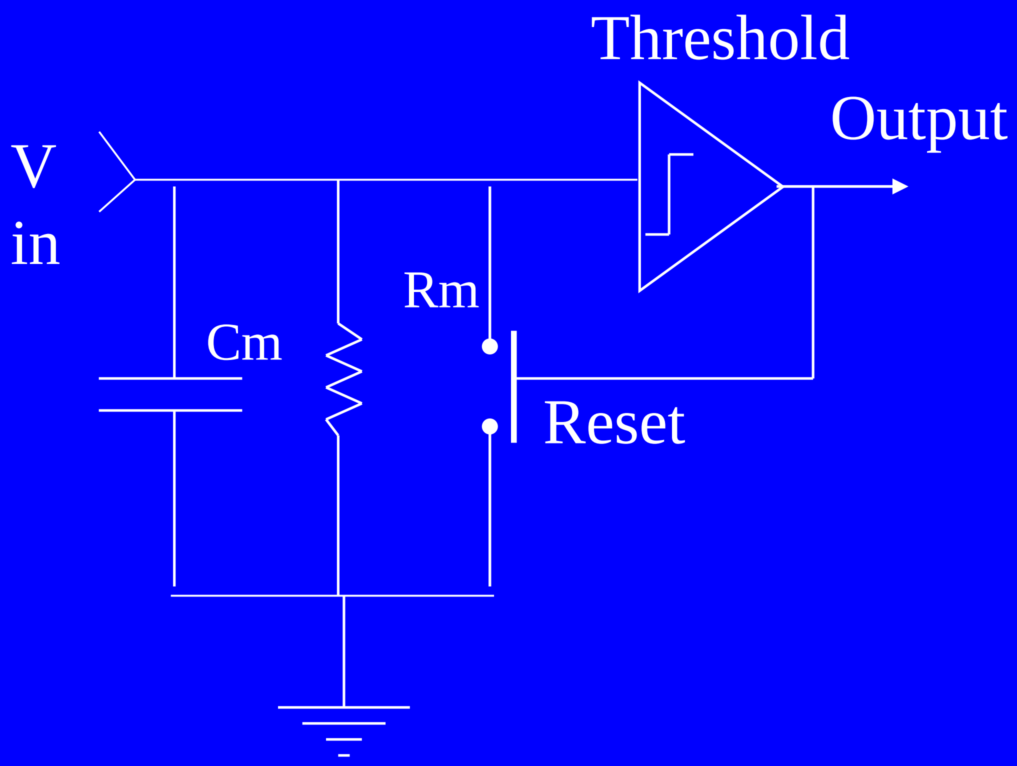
- conductance change over time leading to complex neuronal dynamics.
- However have some constant factors (eg current from pumps) which are grouped together as a *leakage current*.

$$\boxed{\bar{g}_L(V - E_L)}$$

- Over line on g shows that it is constant. Thus it is often called a passive conductance while others termed active conductances

Leaky integrate and fire

- model consists of a capacitor C in parallel with a resistor R driven by a current $I(t)$;
- The driving current can be split into two components, $I(t) = I_R + I_C$.
- The first component is the resistive current I_R which passes through the linear resistor R. (membrane)
- The second component I_C charges the capacitor C
- capacitive current $I_C = C\,\mathrm{d}V/\mathrm{d}t$.



Leaky integrate and fire

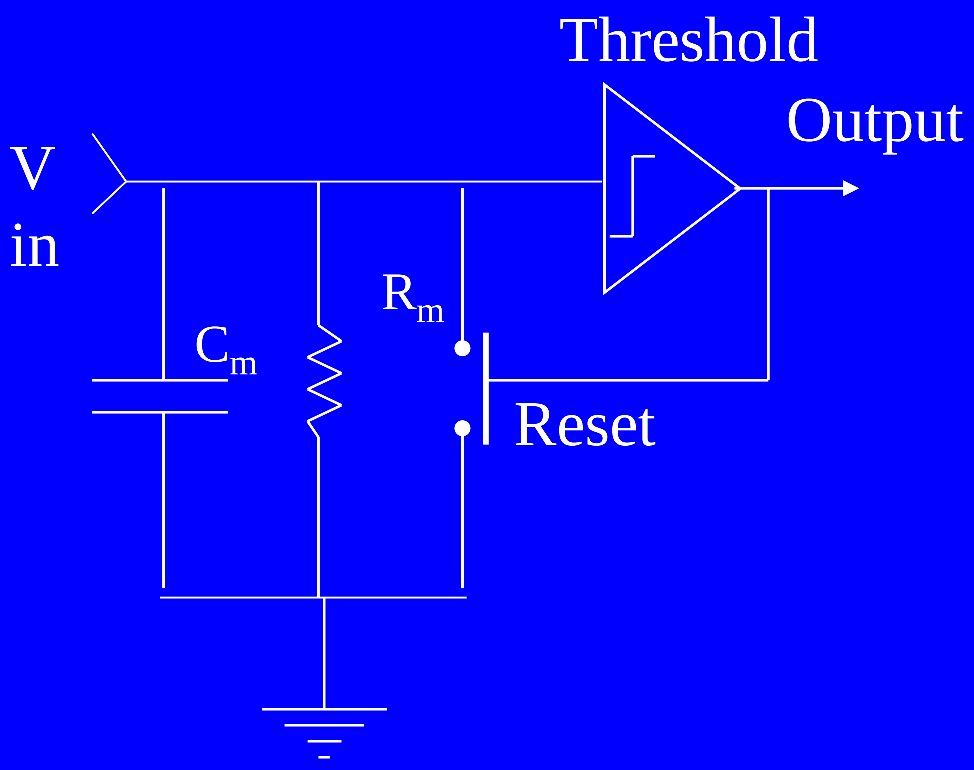
Therefore: $I(t) = I_R + I_C$.

$$I_e(t) = i_R + c_m \frac{dV(t)}{dt}$$

$$I_e = \frac{V}{R_m} + c_m \frac{dV}{dt}$$

Multiplying by R_m we get:

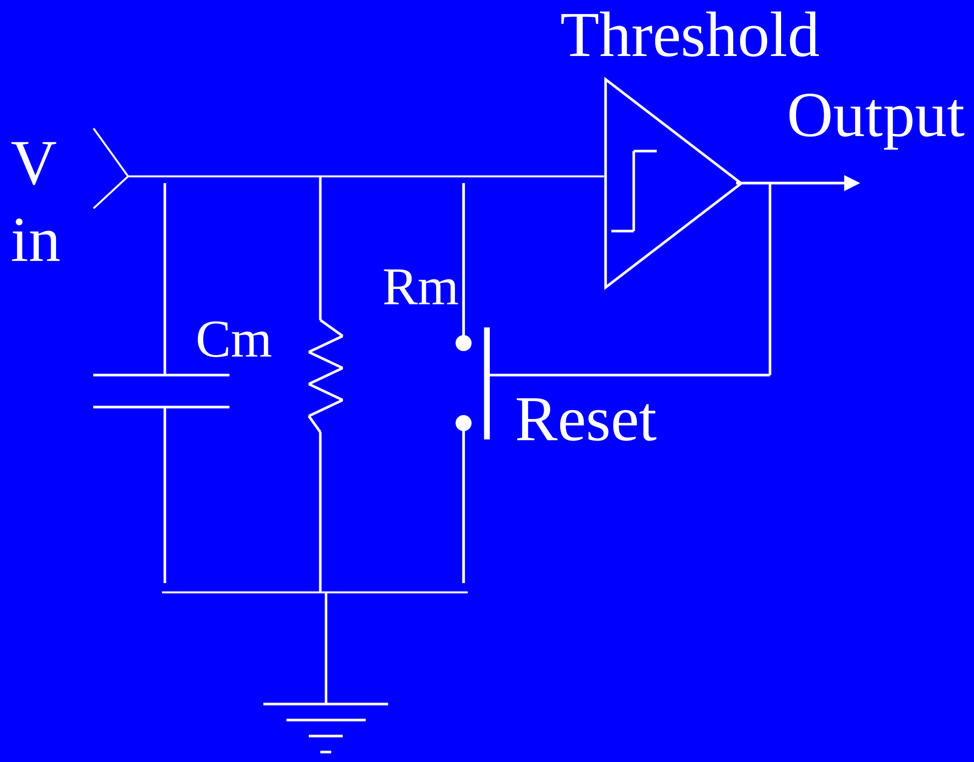
$$R_m I_e(t) = V(t) + \tau_m \frac{dV(t)}{dt}$$

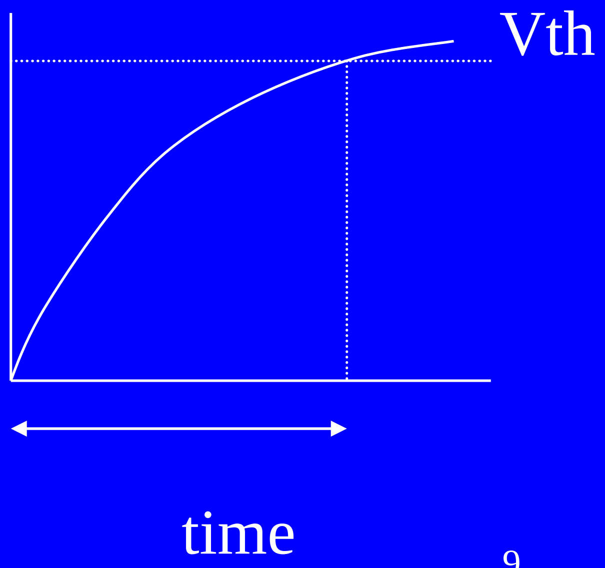


Integrate and Fire Model

If V reaches V_{th} an Action Potential is fired after which V is reset to V_{reset}

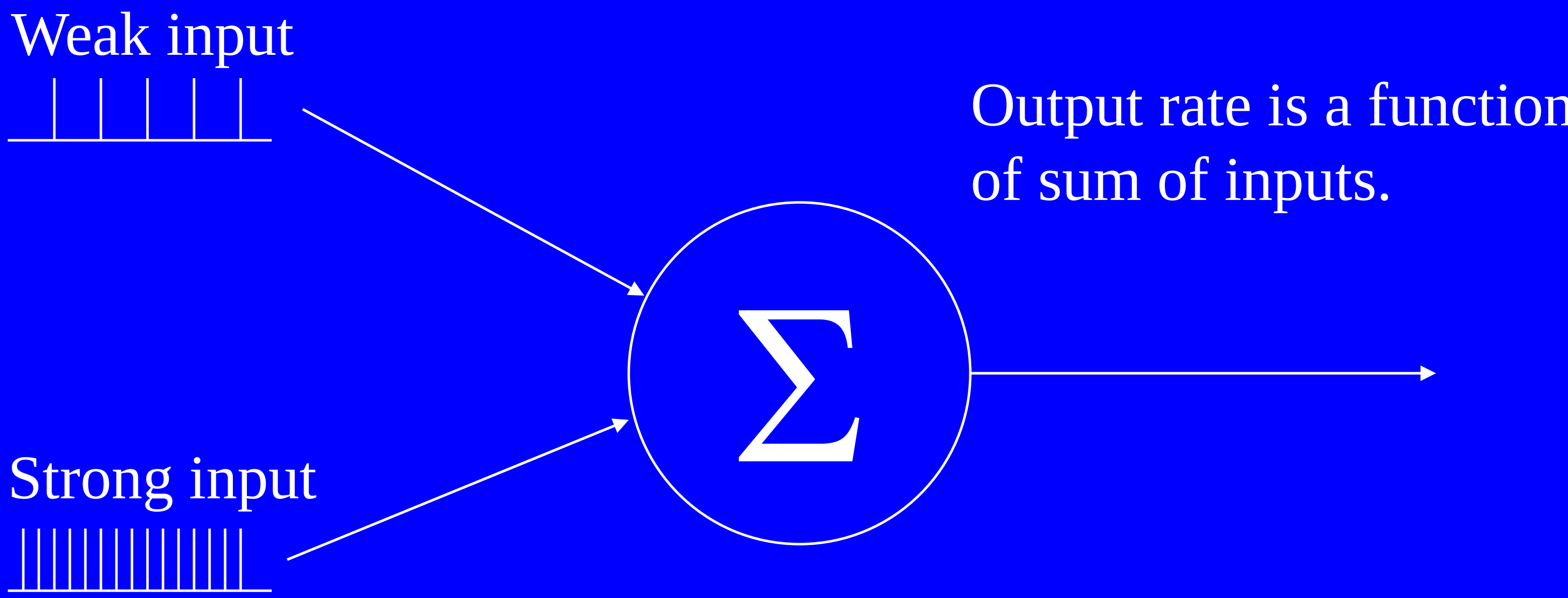
If I_e is 0, V decays exponentially with time constant t_m



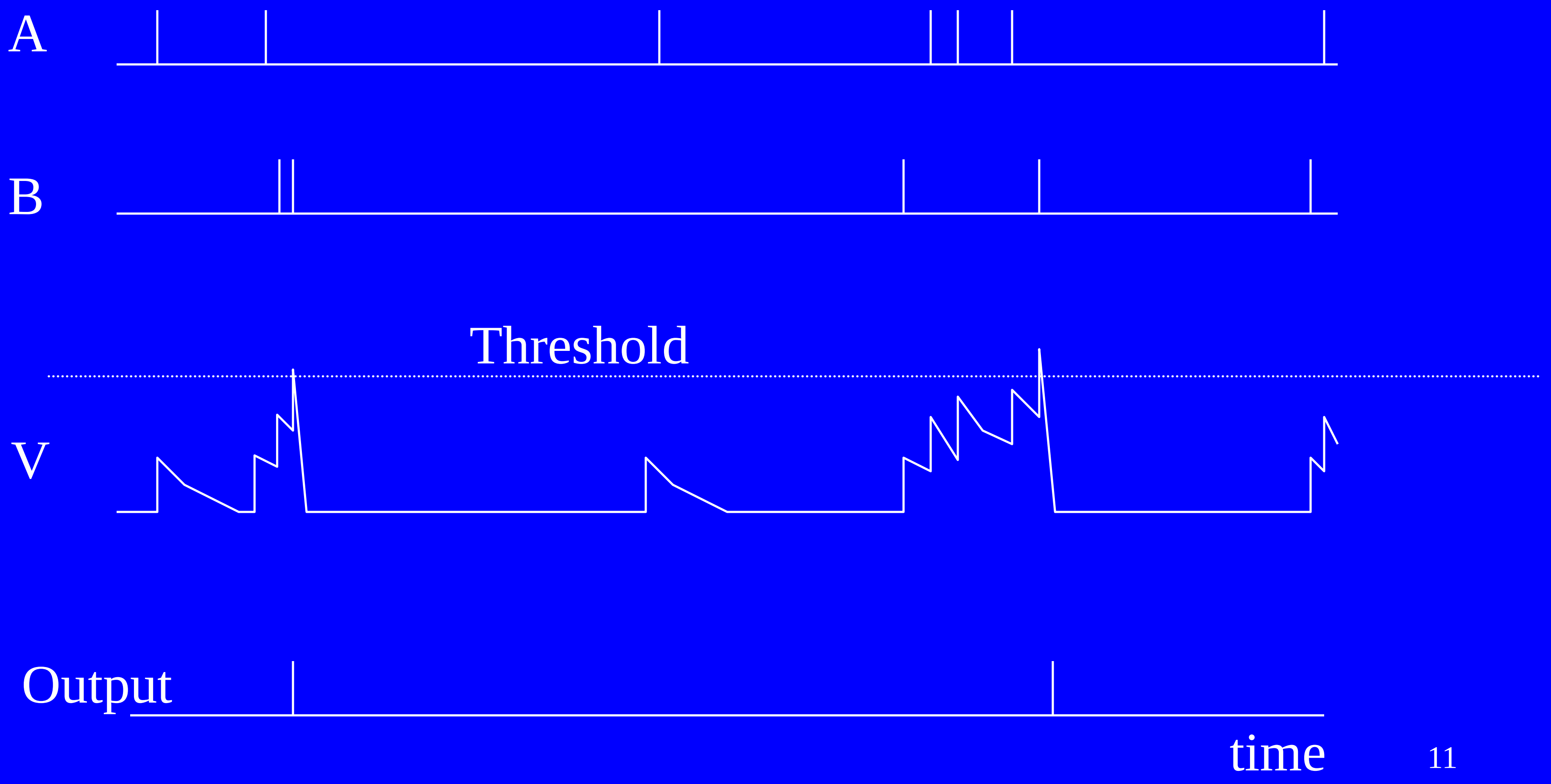


Conceptually similar to ANN neuron

Analog inputs specified by rate



Integrate-and-fire neuron: Summation and resetting



LEARNING

Learning in biological systems

Learning = learning by adaptation

• The young animal learns that the green fruits are sour, while the yellowish/reddish ones are sweet. The learning happens by adapting the fruit picking behavior.

• At the neural level the learning happens by changing of the synaptic strengths, eliminating some synapses, and building new ones.

Learning in BNN

The learning rules of Hebb:

- synchronous activation increases the synaptic strength;
- asynchronous activation decreases the synaptic strength.

Maintaining synaptic strength needs energy

Hebb's LAW

1. If two neurons on either side of a synapse are activated simultaneously (synchronously), then the strength of that synapse is selectively increased.

2. If two neurons on either side of a synapse are activated asynchronously, then that synapse is selectively weakened or eliminated.

Training

- The process of modifying the weights in the connections between network layers with the objective of achieving the expected output is called training a network.

- This is achieved through
  – Supervised learning
  – Unsupervised learning
  – Reinforcement learning

Learning Methods

- Supervised learning
  - Reinforcement learning
  - Corrective learning
- Unsupervised learning
  - Competitive learning
  - Self-organizing learning

Supervised learning

- Teacher: training data
- The teacher scores the performance of the training examples(Always checks whether the out put of NN have reached the target)
- Use performance score to shuffle weights ‘randomly’
- Relatively slow learning due to ‘randomness’

Unsupervised learning

- No help from the outside (No Target input)
- No training data, no information available on the desired output
- *Learning by doing*

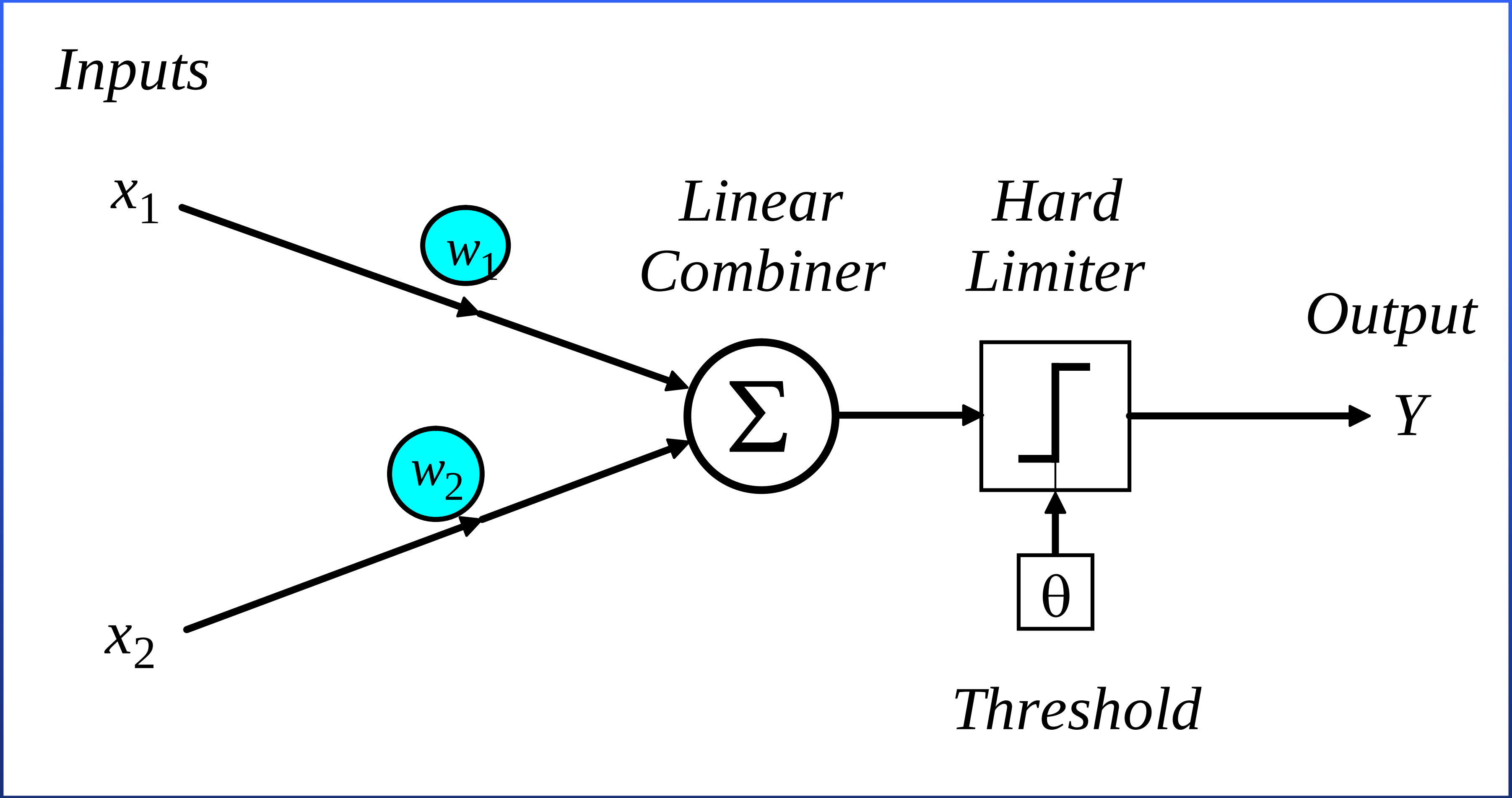
PERCPTRON LEARNING

The Perceptron

Can a single neuron learn a task?

- The perceptron is the simplest form of a neural network. It consists of a single neuron with *adjustable* synaptic weights and a *hard limiter*.
- Network can be trained

Single-layer two-input perceptron



The Perceptron

- The operation of Rosenblatt's perceptron is based on the **McCulloch and Pitts neuron model**. The model consists of a linear combiner followed by a hard limiter.
- The weighted sum of the inputs is applied to the hard limiter, which produces an output equal to +1 if its input is positive and –1 if it is negative.
- Or
- which produces an output equal to +1 if its input is positive and 0 if it is negative.

- The aim of the perceptron is to classify inputs, $x_1, x_2, \ldots, x_n$, into one of two classes, say $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$.

How does the perceptron learn its classification tasks?

The desired output or target out put is presented and is compared with actual output

based on the difference between the actual and desired outputs called error ,the weights are adjusted, to get the desired output of the perceptron.

- If the actual output is o and the desired output is d then the error is given by:

$$e = d - o$$

- If the error, e, is positive, we need to increase perceptron output o, but if it is negative, we need to decrease o.

The perceptron learning rule

$$w_i(p+1) = w_i(p) + \Delta w_i(p)$$

$$w_i(p+1) = w_i(p) + c \cdot x_i(p) \cdot e(p)$$

where p = 1, 2, 3, . . .

c is the **learning rate**, a positive constant less than unity.

.

Perceptron Learning Algorithm

How does the perceptron learn its classification tasks?

The desired output or target out put is presented and is compared with actual output

based on the difference between the actual and desired outputs the weights are adjusted, to get the desired output of the perceptron.

The initial weights are randomly assigned, usually in the range [–0.5, 0.5], and then updated to obtain the output consistent with the training examples.

- If at iteration p, the actual output is $o(p)$ and the desired output is $d(p)$, then the error is given by:

$$e(p) = d(p) - o(p)$$

where p = 1, 2, 3, . . .

Iteration p here refers to the pth training example presented to the perceptron.

- If the error, $e(p)$, is positive, we need to increase perceptron output $o(p)$, but if it is negative, we need to decrease $o(p)$.

The perceptron learning rule

$$w_i(p+1) = w_i(p) + c \cdot x_i(p) \cdot e(p)$$

where p = 1, 2, 3, . . .

c is the **learning rate**, a positive constant less than unity.

The perceptron learning rule was first proposed by **Rosenblatt** in 1960. Using this rule we can derive the perceptron training algorithm for classification tasks.

Perceptron's training algorithm
Discrete perceptron

Step 1: Initialisation

Set initial weights $w_1, w_2, \ldots, w_n$ and threshold θ to random numbers in the range [–0.5, 0.5].

If the error, $e(p)$, is positive, we need to increase perceptron output $o(p)$, but if it is negative, we need to decrease $o(p)$.

Perceptron's training algorithm (continued)

Step 2: Activation

Activate the perceptron by applying inputs $x_1(p)$, $x_2(p)$,…, $x_n(p)$. Calculate the actual output at iteration $p = 1$

$$o(p) = step\left[\sum_{i=1}^{n} x_i(p)\, w_i(p) - \theta\right]$$

where n is the number of the perceptron inputs, and *step* is a step activation function.

In place of step function for continuous neuron any other activation functions can be used

Perceptron's training algorithm (continued)

- At iteration p, the actual output is $o(p)$ and the desired output is $d(p)$, then calculate error by:

$$e(p) = d(p) - o(p)$$

where p = 1, 2, 3, . . .

Iteration p here refers to the pth training example presented to the perceptron.

Perceptron's training algorithm (continued)

Step 3: Weight training

Update the weights of the perceptron

$$w_i(p+1) = w_i(p) + \Delta w_i(p)$$

where $\Delta w_i(p)$ is the weight correction at iteration p.

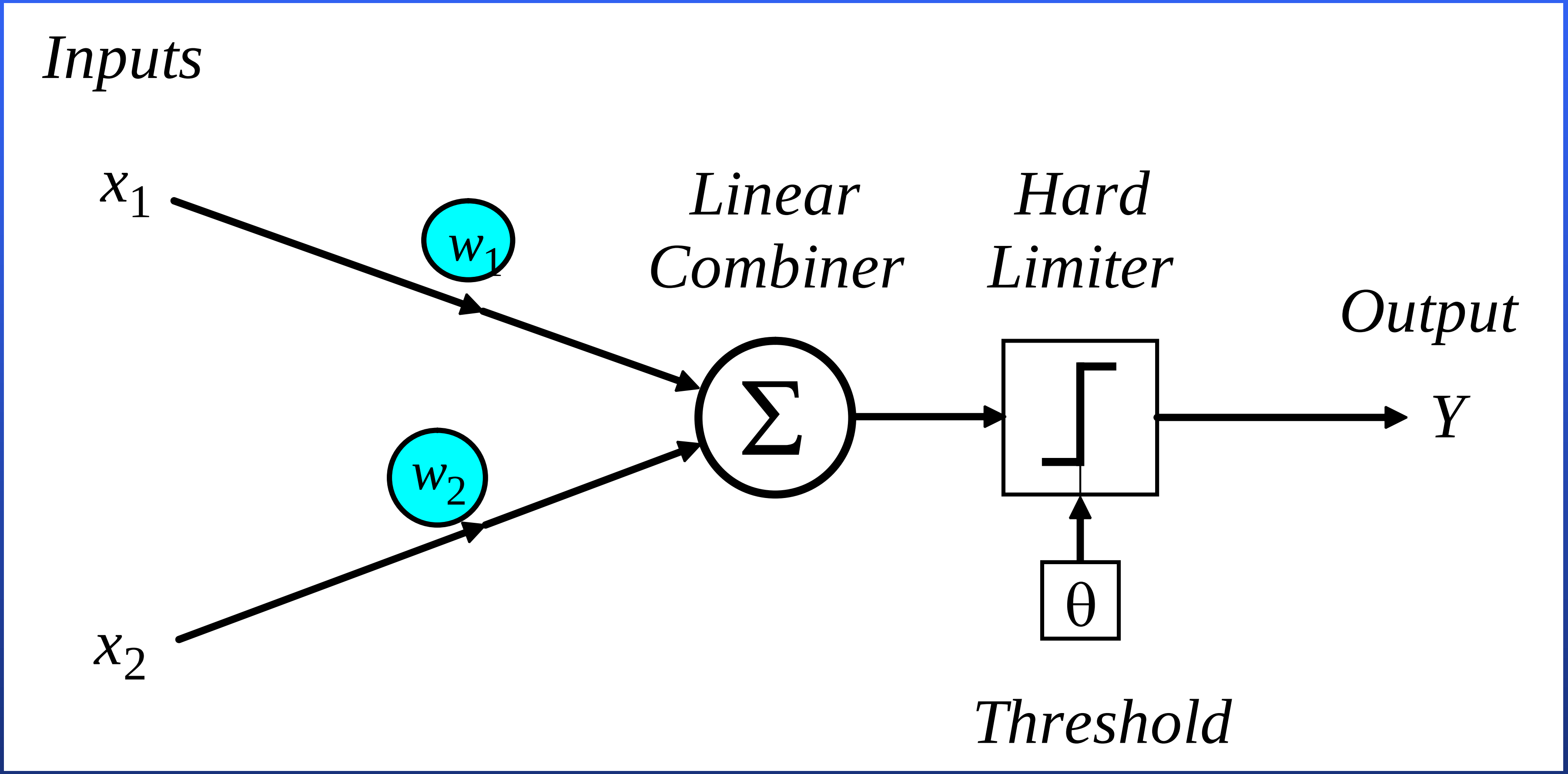
The weight correction is computed by the **delta rule (perceptron learning rule)**

$$\boxed{\Delta w_i(p) = c \cdot x_i(p) \cdot e(p)}$$

Step 4: Iteration

Increase iteration p by one, go back to *Step 2* and repeat the process until convergence.

Example



Inputs
x_1
w_1
x_2
w_2
Linear Combiner
Σ
Hard Limiter
Output
Y
θ
Threshold

- initialisation:
  $w_1 = 0.3, \quad w_2 = -\,0.1, \quad \theta = 0.2.$

$$o(p) = step\left[\sum_{i=1}^{n} x_i(p)\,.w_i(p) - \theta\right]$$

$$o(p) = step\left[x_1(p)\,.w_1(p) + x_2(p)\,.w_2(p) - \theta\right]$$

- input x(1) = [0 0]

$$o(1) = step\left[0\,.(0.3) + 0.(-0.1) - 0.2 = -0.2\right] = 0$$

Error = d – o = 0

Epoch	Inputs		Desired output
	x_1	x_2	Y_d
1	0	0	0
	0	1	0
	1	0	0
	1	1	1

Change in weight $\Delta w_i(p) = c \cdot x_i(p) \cdot e(p)$

$$\Delta w_1(1) = c \cdot x_1(1) \cdot e(1) = 0.1.(0).(0) = 0$$
$$\Delta w_2(1) = c \cdot x_2(1) \cdot e(1) = 0.1.(0).(0) = 0$$

New weight

$$w_i(p+1) = w_i(p) + \Delta w_i(p)$$

$$w_1(2) = w_1(1) + \Delta w_1(1) = 0.3 + 0 = 0.3$$
$$w_2(2) = w_2(1) + \Delta w_2(1) = -0.1 + 0 = -0.1$$

Epoch	Inputs		Desired output
	x_1	x_2	Y_d
1	0	0	0
	0	1	0
	1	0	0
	1	1	1

p = 2 present x(2) = [0 1]

$$o(2) = step[x1(2).w1(2) + x2(2).w2(2) = 0.(0.3) + 1.(-0.1) - 0.2 = -0.3] = 0$$

Error e(2) = 0 -0 =0

Change in weight

$$\Delta w_1(2) = c \cdot x_1(2) \cdot e(2) = 0.1.(0).(0) = 0$$

$$\Delta w_2(2) = c \cdot x_2(2) \cdot e(2) = 0.1.(1).(0) = 0$$

Epoch	Inputs x_1	Inputs x_2	Desired output Y_d
1	0	0	0
	0	1	0
	1	0	0
	1	1	1

New weight

$$w_1(3) = w_1(2) + \Delta w_1(2) = 0.3 + 0 = 0.3$$

$$w_2(3) = w_2(2) + \Delta w_2(2) = -0.1 + 0 = -0.1$$

p = 3 present x(3) = [1 0]

$$o(3) = step\left[x1(3).w1(3) + x2(3).w2(3) = 1.(0.3) + 0.(-0.1) - 0.2 = 0.1\right] = 1$$

Error e(3) = 0 -1 = -1

Change in weight

$$\Delta w_1(3) = c \cdot x_1(3) \cdot e(3) = 0.1.(1).(-1) = -0.1$$

$$\Delta w_2(3) = c \cdot x_2(3) \cdot e(3) = 0.1.(0).(-1) = 0$$

Epoch	Inputs x_1	Inputs x_2	Desired output Y_d
1	0	0	0
	0	1	0
	1	0	0
	1	1	1

New weight

$$w_1(4) = w_1(3) + \Delta w_1(3) = 0.3 - 0.1 = 0.2$$

$$w_2(4) = w_2(3) + \Delta w_2(3) = -0.1 + 0 = -0.1$$

p = 4 present x(4) = [1 1]

$$o(4) = step[x1(4).w1(4) + x2(4).w2(4) = 1.(0.2) + 1.(-0.1) - 0.2 = -0.1] = 0$$

Error e(4) = 1 -0 = 1

Change in weight

$$\Delta w_1(4) = c \cdot x_1(4) \cdot e(4) = 0.1.(1).(1) = 0.1$$

$$\Delta w_2(4) = c \cdot x_2(4) \cdot e(4) = 0.1.(1).(1) = 0.1$$

Epoch	Inputs x_1	Inputs x_2	Desired output Y_d
1	0	0	0
	0	1	0
	1	0	0
	1	1	1

New weight

$$w_1(5) = w_1(4) + \Delta w_1(4) = 0.2 + 0.1 = 0.3$$

$$w_2(5) = w_2(4) + \Delta w_2(4) = -0.1 + 0.1 = 0$$

Example of perceptron learning: the logical operation *AND*

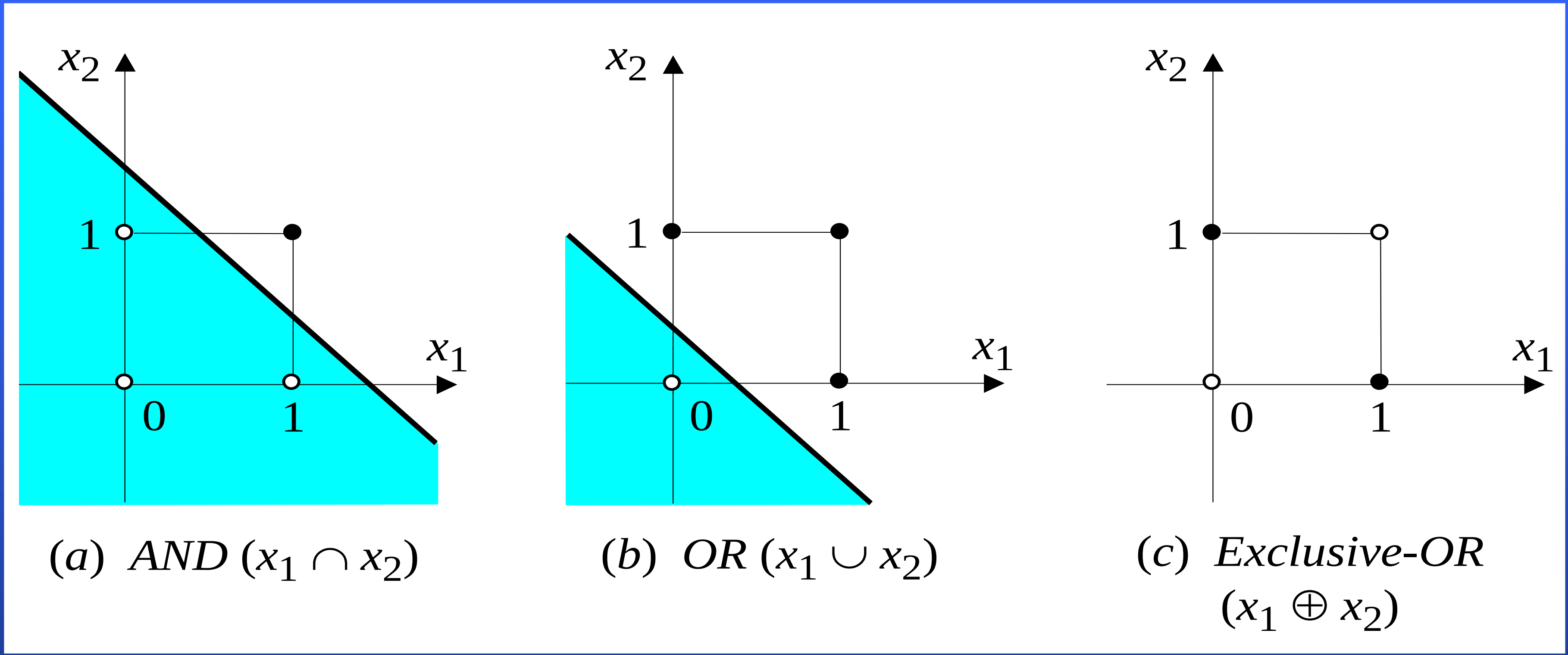
Epoch	Inputs		Desired output	Initial weights		Actual output	Error	Final weights	
	x_1	x_2	Y_d	w_1	w_2	Y	e	w_1	w_2
1	0	0	0	0.3	−0.1	0	0	0.3	−0.1
	0	1	0	0.3	−0.1	0	0	0.3	−0.1
	1	0	0	0.3	−0.1	1	−1	0.2	−0.1
	1	1	1	0.2	−0.1	0	1	0.3	0.0

Example of perceptron learning: the logical operation *AND*

Epoch	Inputs		Desired output	Initial weights		Actual output	Error	Final weights	
	x_1	x_2	Y_d	w_1	w_2	Y	e	w_1	w_2
1	0	0	0	0.3	–0.1	0	0	0.3	–0.1
	0	1	0	0.3	–0.1	0	0	0.3	–0.1
	1	0	0	0.3	–0.1	1	–1	0.2	–0.1
	1	1	1	0.2	–0.1	0	1	0.3	0.0
2	0	0	0	0.3	0.0	0	0	0.3	0.0
	0	1	0	0.3	0.0	0	0	0.3	0.0
	1	0	0	0.3	0.0	1	–1	0.2	0.0
	1	1	1	0.2	0.0	1	0	0.2	0.0
3	0	0	0	0.2	0.0	0	0	0.2	0.0
	0	1	0	0.2	0.0	0	0	0.2	0.0
	1	0	0	0.2	0.0	1	–1	0.1	0.0
	1	1	1	0.1	0.0	0	1	0.2	0.1
4	0	0	0	0.2	0.1	0	0	0.2	0.1
	0	1	0	0.2	0.1	0	0	0.2	0.1
	1	0	0	0.2	0.1	1	–1	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1
5	0	0	0	0.1	0.1	0	0	0.1	0.1
	0	1	0	0.1	0.1	0	0	0.1	0.1
	1	0	0	0.1	0.1	0	0	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1

Threshold: $\theta = 0.2$; learning rate: $\alpha = 0.1$

Two-dimensional plots of basic logical operations



A perceptron can learn the operations *AND* and *OR*, but not *Exclusive-OR*.