An Introduction to Neural Computing

CMP9783 – Neural Computing

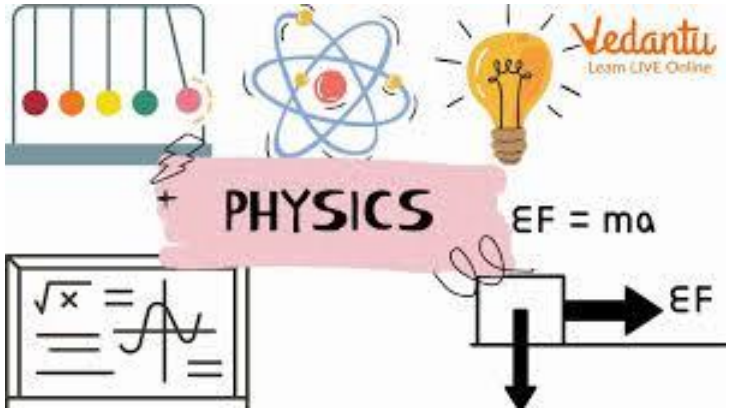
Week 2

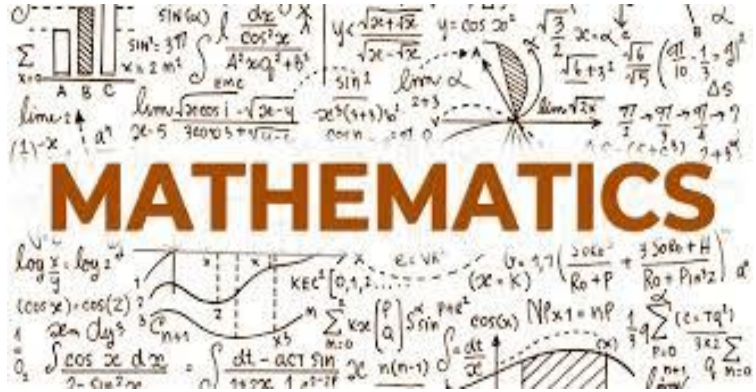
Dr Christos Frantzidis
Senior Lecturer in Computer Science – Machine Learning
cfrantzidis@lincoln.ac.uk

Please evaluate the lecture at the following link:

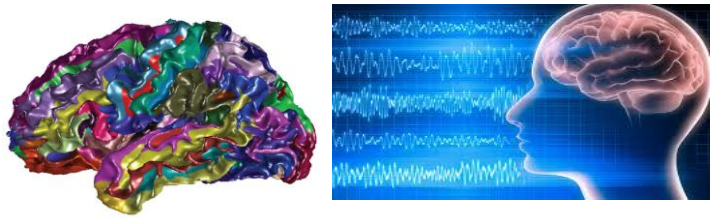
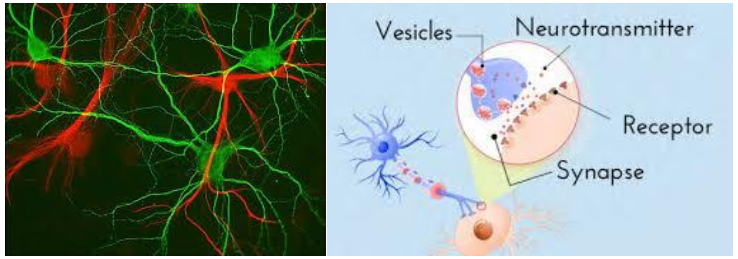
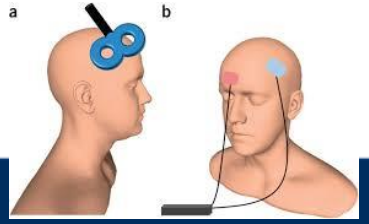
What is computational neuroscience?

It is a field that combines elements of neuroscience, computer science, mathematics and physics to understand how the brain processes information.





A mechanistic understanding of the brain

Domain	Methodology	Objective
A. Neuroanatomy & Neurophysiology		To identify brain structures and understand their role
B. Molecular & Cellular Neuroscience		To explore how individual neurons and synapses function at a molecular
C. Circuitry & Network Dynamics		To understand how groups of neurons interact to form circuits and networks that process information
D. Behavioral & Systems Neuroscience		To relate specific neural mechanisms to behaviors and cognitive functions

What is a model ???

1. Mathematical Formula

$$\tau_m \frac{dV(t)}{dt} = -(V(t) - V_{\text{rest}}) + R_m I(t)$$

The Leaky Integrate-and-Fire (LIF) model

- One of the simplest and most widely used neuron models
- It is described by a single differential equation
- It captures the dynamics of a neuron's membrane potential.

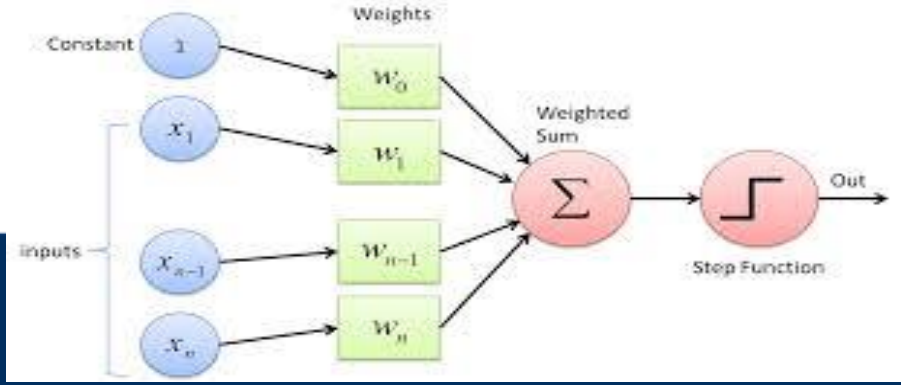
- The equation describes how the membrane potential changes over time in response to input current.
- When the membrane potential V(t) reaches a certain threshold, the neuron "fires" and then resets to a resting state.

2. Computer Code

```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters for the LIF neuron
tau_m = 20.0  # Membrane time constant (ms)
R_m = 1.0     # Membrane resistance (MΩ)
V_rest = -65  # Resting membrane potential (mV)
V_th = -50    # Firing threshold (mV)
V_reset = -65 # Reset potential (mV) after spike
I_input = 1.5 # Input current (nA)
t_sim = 200   # Total simulation time (ms)
dt = 0.1      # Time step (ms)

# Time array
time = np.arange(0, t_sim, dt)

# Array to store membrane potential over time
V_m = np.zeros(len(time))
V_m[0] = V_rest  # Initialize the membrane potential at rest

# Array to store spikes
spikes = np.zeros(len(time))

# Simulation of the LIF neuron
for t in range(1, len(time)):
    dV = (-(V_m[t-1] - V_rest) + R_m * I_input) * (dt / tau_m)
    V_m[t] = V_m[t-1] + dV  # Update membrane potential

    # Check if the neuron fires a spike
    if V_m[t] >= V_th:
        V_m[t] = V_reset  # Reset the membrane potential
        spikes[t] = 1     # Record the spike event

# Plot membrane potential over time
plt.figure(figsize=(10, 6))
plt.plot(time, V_m, label='Membrane Potential (mV)', color='b')
plt.axhline(y=V_th, color='r', linestyle='--', label=f'Threshold: {V_th} mV')
plt.title('Leaky Integrate-and-Fire Neuron Model')
plt.xlabel('Time (ms)')
plt.ylabel('Membrane Potential (mV)')
plt.legend()
plt.grid(True)
plt.show()

# Plot spikes over time
plt.figure(figsize=(10, 2))
plt.plot(time, spikes, color='g', label='Spikes', marker='|', linestyle='None')
plt.title('Neuron Spikes')
plt.xlabel('Time (ms)')
plt.ylabel('Spikes')
plt.ylim(-0.1, 1.1)
plt.grid(True)
plt.show()
```

3. Hypothetical System

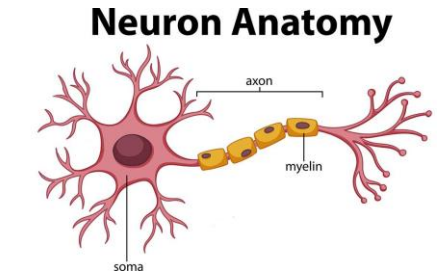
- Hypothetical systems are often idealized or abstract representations of neural processes.
- They focus on key principles or functions rather than biological details.
- The perceptron is a hypothetical neuron that computes a weighted sum of its inputs and passes it through a step function.
- It learns to adjust its weights to minimize classification errors based on a simple learning rule.

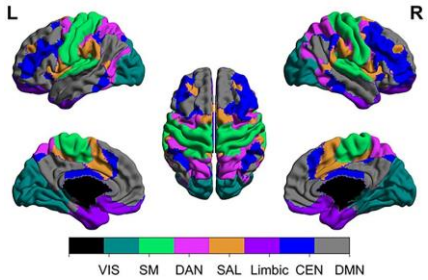


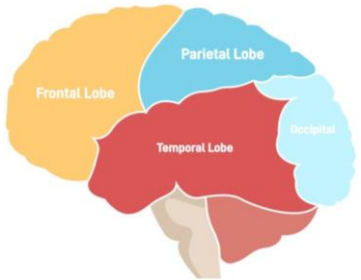
LINCOLN

Models in Computational Neuroscience

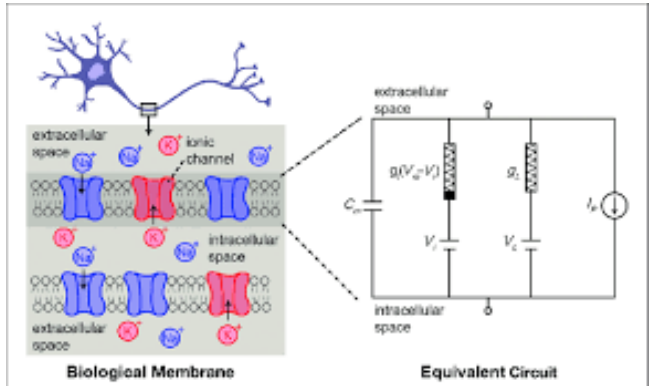
Neural System

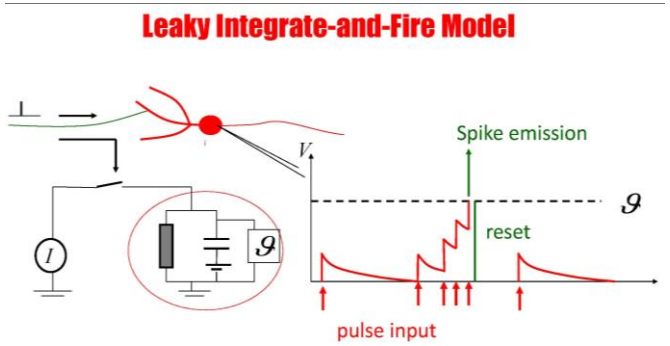






Mathematical Representation





Model's Aim

explain

simulate

Model vs. Hypothesis

Scientists develop hypotheses of the underlying mechanisms of a system that have to be tested against reality

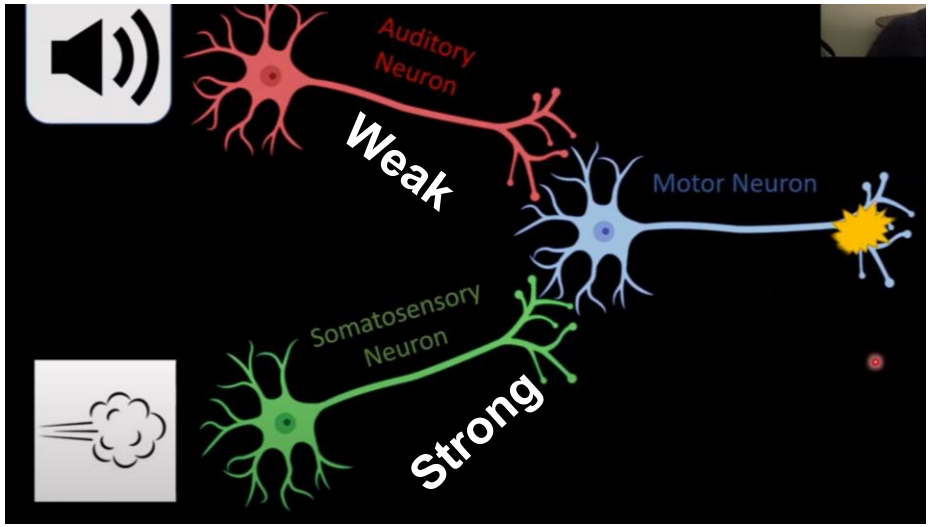
To test a specific feature of a hypothesis, we build a model that can be evaluated

Hebbian Plasticity Hypothesis:

"Cells that fire together, wire together"

Long Term Potentiation (LTP) Model

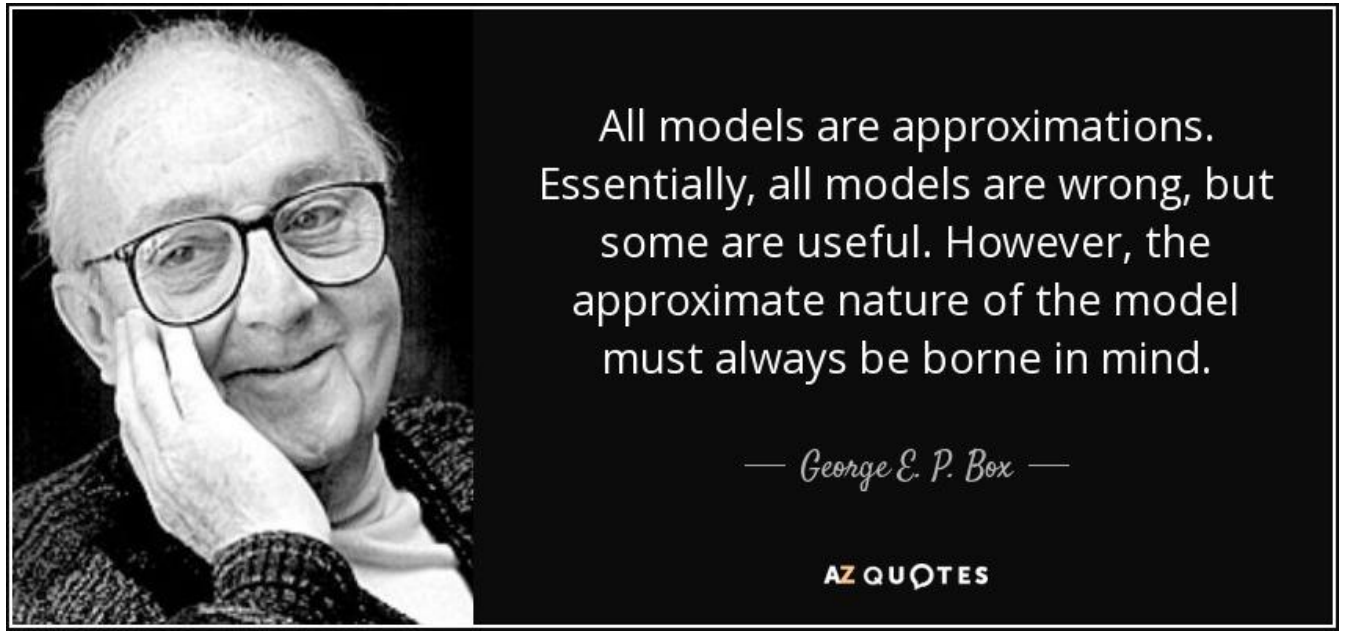
The synaptic strength is increased after a period of intense, coordinated activity between neurons



- The hypothesis (Hebbian plasticity) is formed based on theoretical considerations about how learning might work in the brain.
- Experiments testing LTP and synaptic strength usually provide evidence supporting the hypothesis.
- The reality is that additional mechanisms, beyond what was originally submitted, are involved in synaptic plasticity!!!

Model Definitions

1. A model is an abstraction of a real-world system to demonstrate particular features of, or investigate specific questions about, the system.
2. A model is a quantification of a hypothesis to investigate the hypothesis.

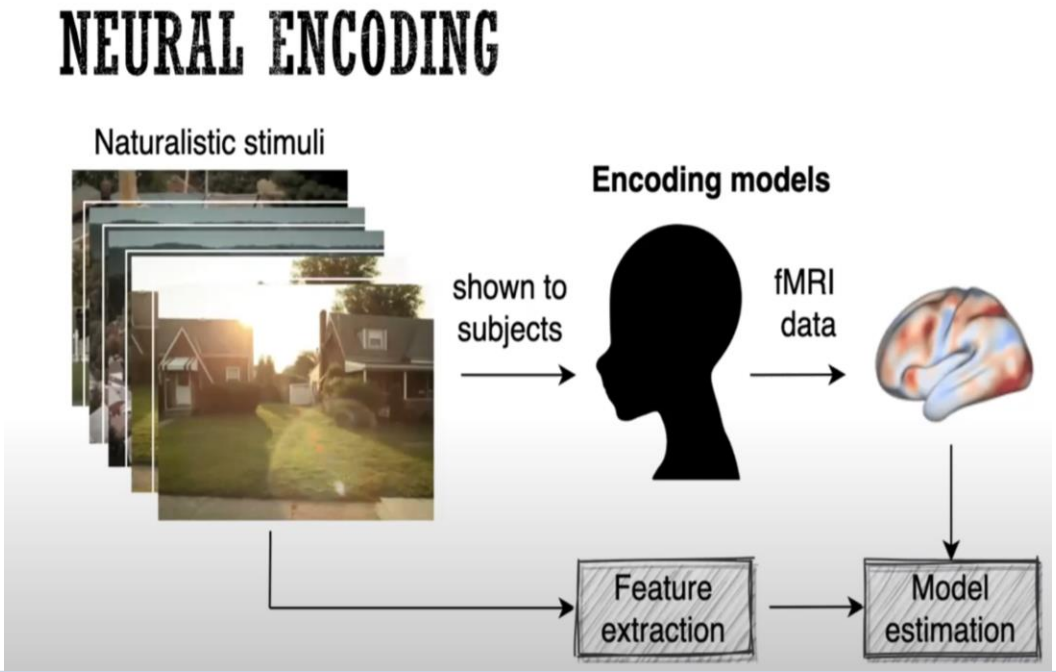


How to use models (1)

A neural encoding model

Apart from studying specific aspects of a hypothesis or theory, models can also help to interpret experimental data.

- fMRI data provide researchers with information about which brain regions are active while participants engage in tasks, such as viewing images or listening to sounds.
- fMRI signals reflect blood oxygenation levels rather than actual neural firing.
- This is an indirect measure of neural activity.
- To interpret these signals more meaningfully, researchers often use neural encoding models.



The neural encoding model: Methodological Approach

1. Hypothesis

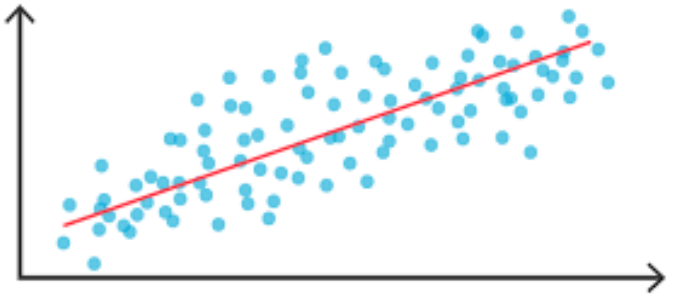
Researchers may hypothesize that certain brain regions are responsible for encoding specific types of visual information, such as faces, shapes or movement.

2. Experimental Data Collection

- We design an fMRI experiment where participants view different types of images
- We collect brain activity data during the task.
- The raw fMRI data show increased activation in certain brain regions.
- The exact nature of the information encoded in these regions remains unclear without further analysis.

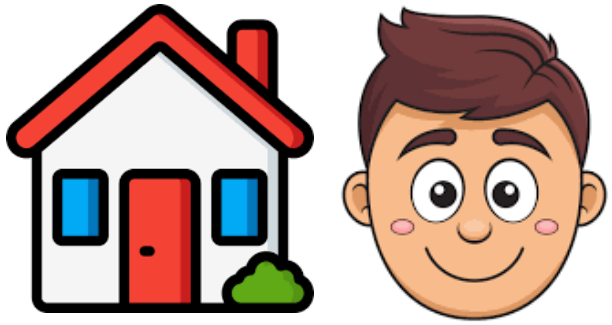
3. Neural Encoding Model

- It predicts how specific patterns of neural activity relate to the stimuli properties
- For example, the model predicts that brain region X responds more strongly to images of faces, while brain region Y responds to abstract patterns
- By comparing the model's predictions to the actual fMRI data, researchers can test whether the activity in different brain regions correlates with specific features of the visual stimuli.

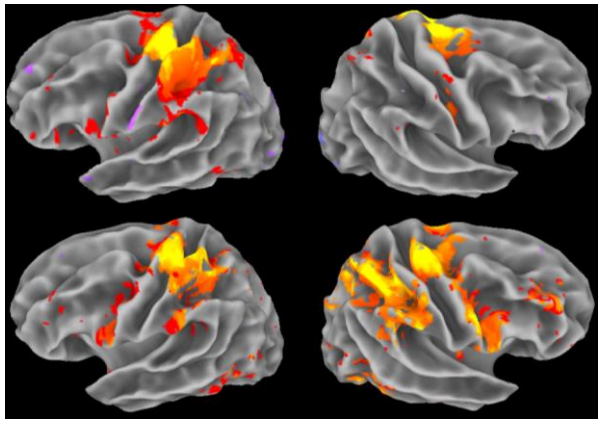


The neural encoding model: The cookbook

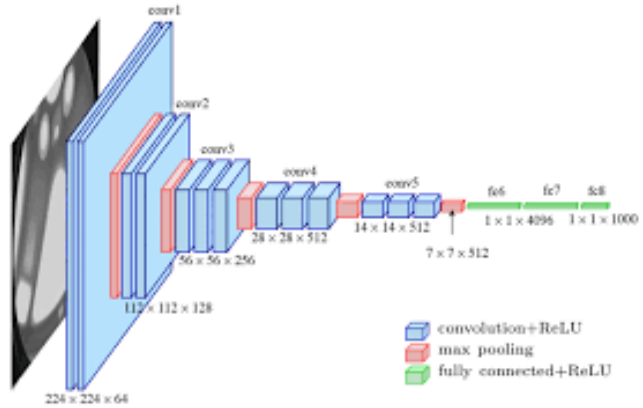
1. Stimuli



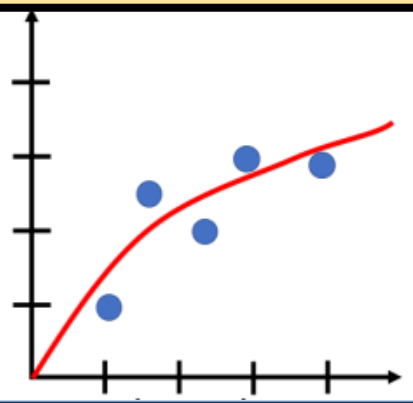
2. Data Collection



3. Feature Extraction



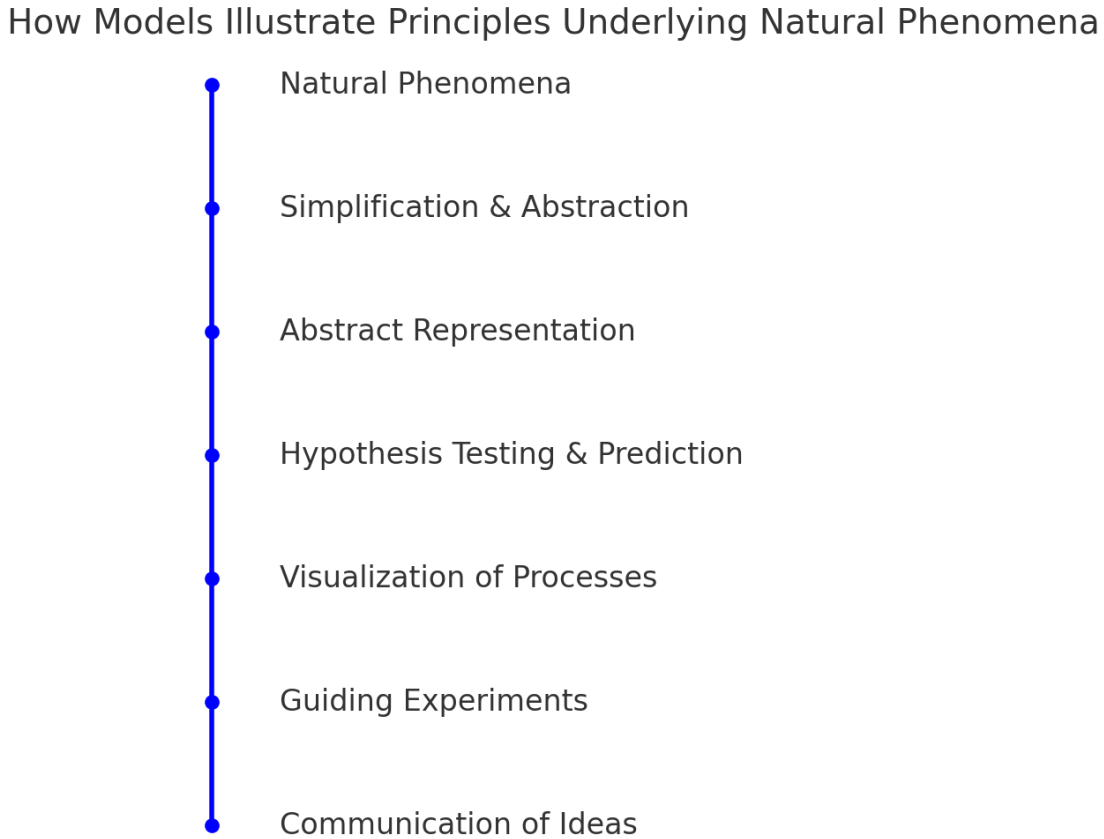
4. Encoding Model



5. Prediction of unknown data

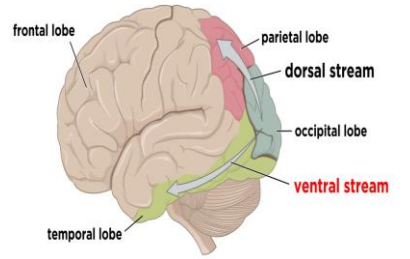
How to use models (1)

Models illustrate principles underlying natural phenomena on a conceptual level.



Simplification of complexity

Natural Phenomenon: The brain's processing of visual information involves billions of neurons and intricate networks.



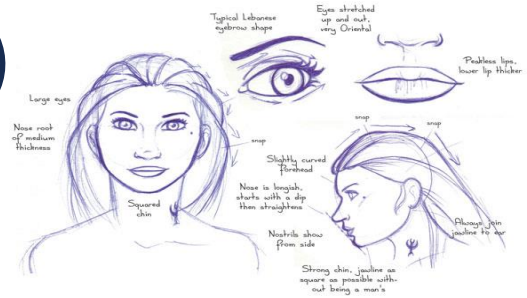
Visual inputs like faces or patterns

1. Stimuli

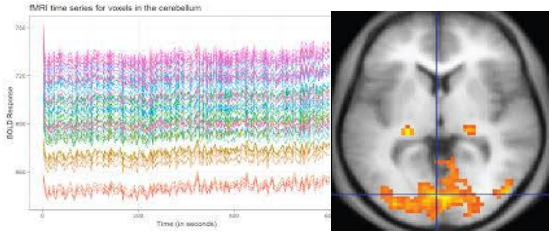
Simplified Neural Encoding Model

Breaking down stimuli into measurable features

2. Feature Extraction

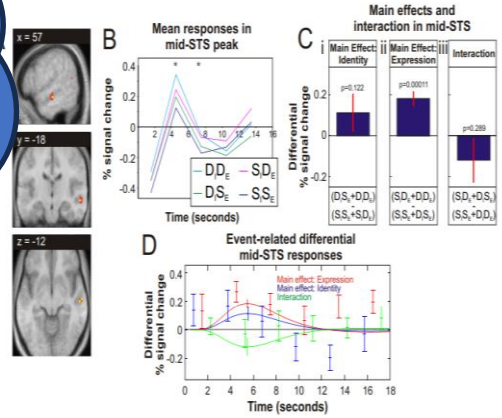


3. Brain Activity



Establishing relationships between features and neural responses

4. Brain



By reducing complexity, the model allows us to grasp how specific features of stimuli are represented in the brain.

Neural responses computed by fMRI data

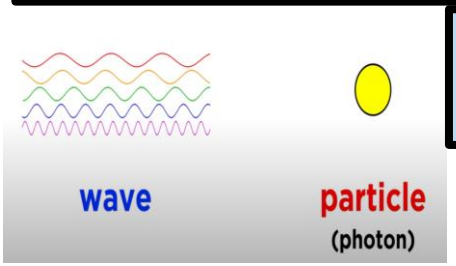
UNIVERSITY OF LINCOLN

https://www.youtube.com/watch?v=MgMNUne9j9

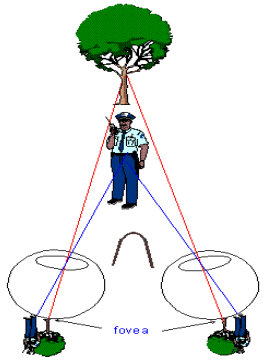
Winston, J.S., Henson, R.N.A., Fine-Goulden, M.R. and Dolan, R.J., 2004. fMRI-adaptation reveals dissociable neural representations of identity and expression in face perception. *Journal of neurophysiology*, *92*(3), pp.1830-1839.

Abstract representation of principles

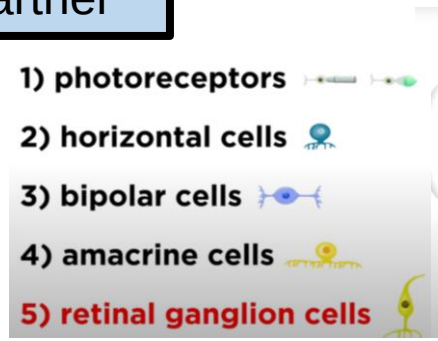
Natural Principle: The brain encodes and processes sensory information to generate perception.

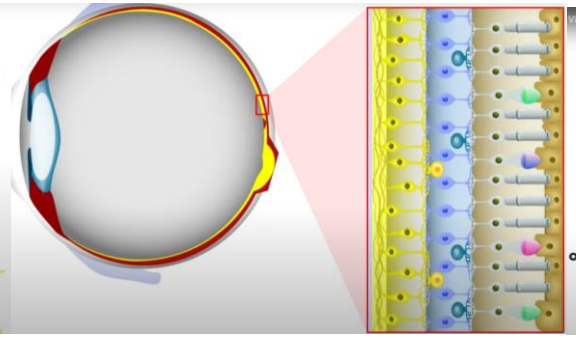


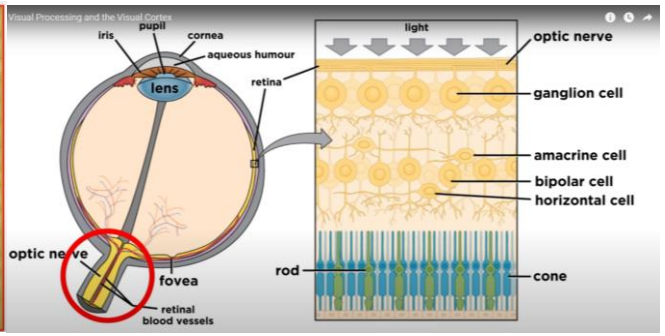
Greater disparity = Closer
Lesser disparity = Farther

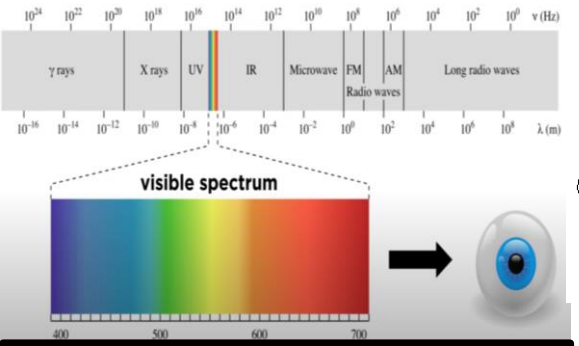


1. Feature Extraction

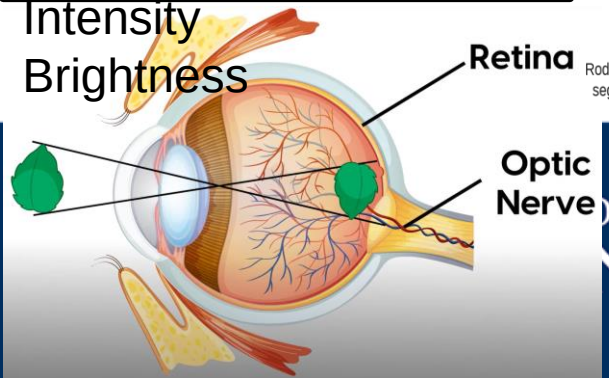


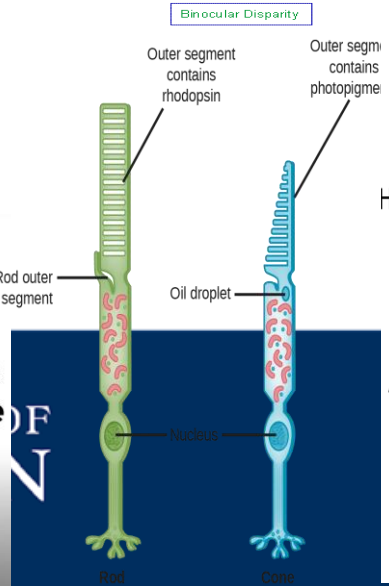


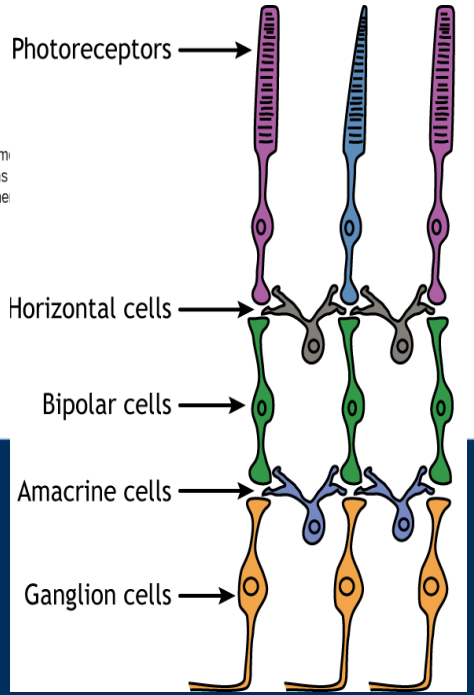




Wavelength
Color
Intensity
Brightness







Rods & Cones

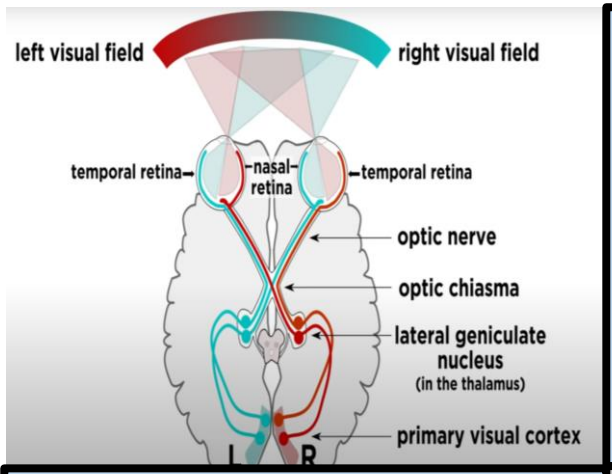
Light ➡ Electric signal

- Rodes are more efficient in low-light condition
- Cones detect color in brighter conditions
- Signals are then passed to retinal ganglion

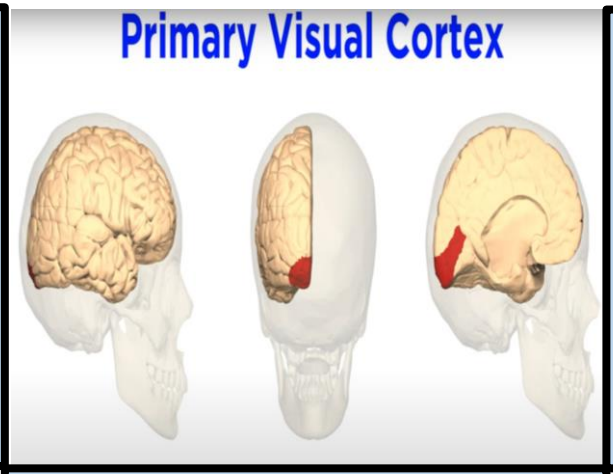
Retinal Ganglion Cells

- First neurons to start feature extraction
- Detect basic patterns of light & dark
- The ganglion cells then send these early visual signals through the optic nerve to the brain

Feature Extraction in detail

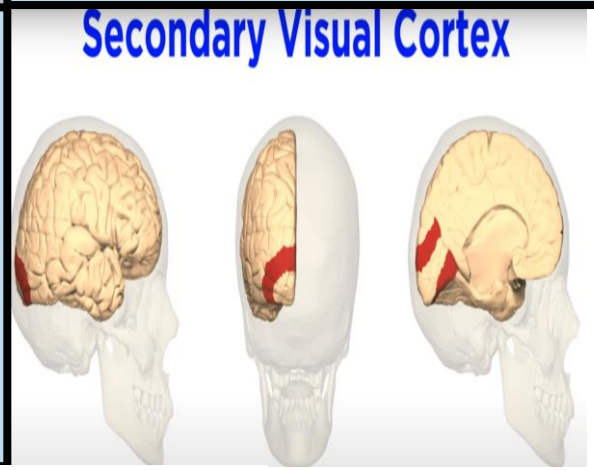


- The signals from the retina travel to the lateral geniculate nucleus (LGN)
- The LGN organizes information from both eyes and filters the visual input based on its importance.

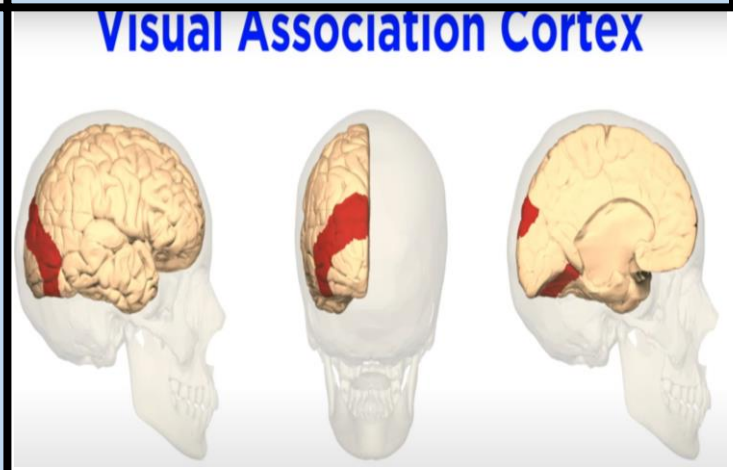


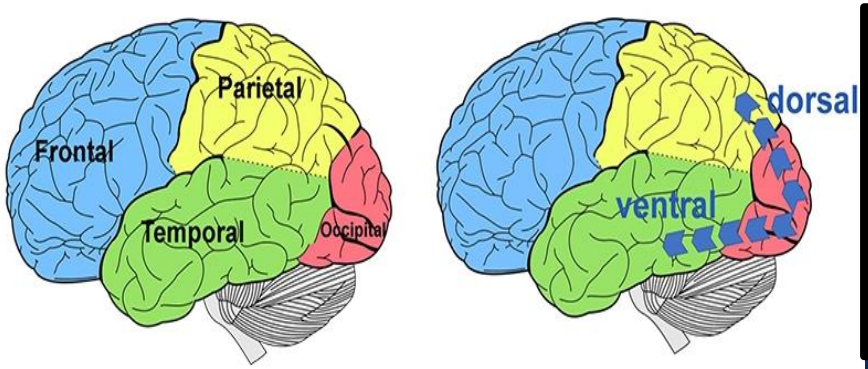
- It is the first stage of detailed feature extraction.
- V1 neurons are organized in a way that allows them to detect edges, orientations and spatial frequencies
- Understanding of the scene structure

- V2 neurons are responsible for detecting more complex patterns like angles or junctions where edges meet.
- This area also processes texture and more intricate details.



- It integrates and interprets the basic visual signals processed by V1, allowing us to recognize and understand faces, objects, scenes.
- It plays role in visual





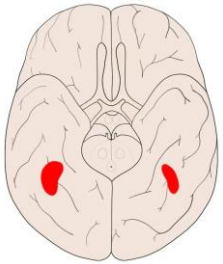
- The dorsal stream processes information related to motion and spatial location.
- It helps the brain to understand where objects are in space and how to interact with them.
- It processes information related to object identity and recognition.
- This pathway helps identify what the objects are (object's shape, color, texture, identity)

Hypothesis Testing & prediction

Natural Inquiry: How does the brain differentiate between faces and abstract patterns?

1. Hypothesis Formulation

Is the Fusiform Face Area (FFA) a specialized brain region for face recognition?



2. Prediction

Higher FFA activity when participants show faces

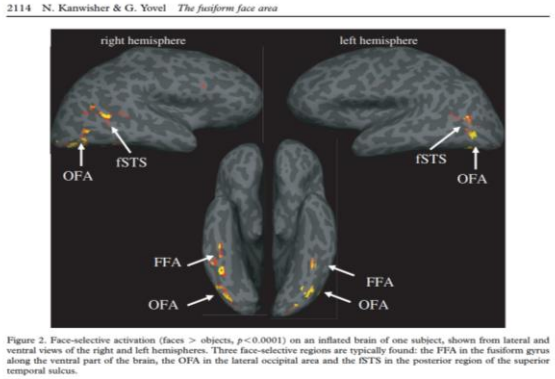
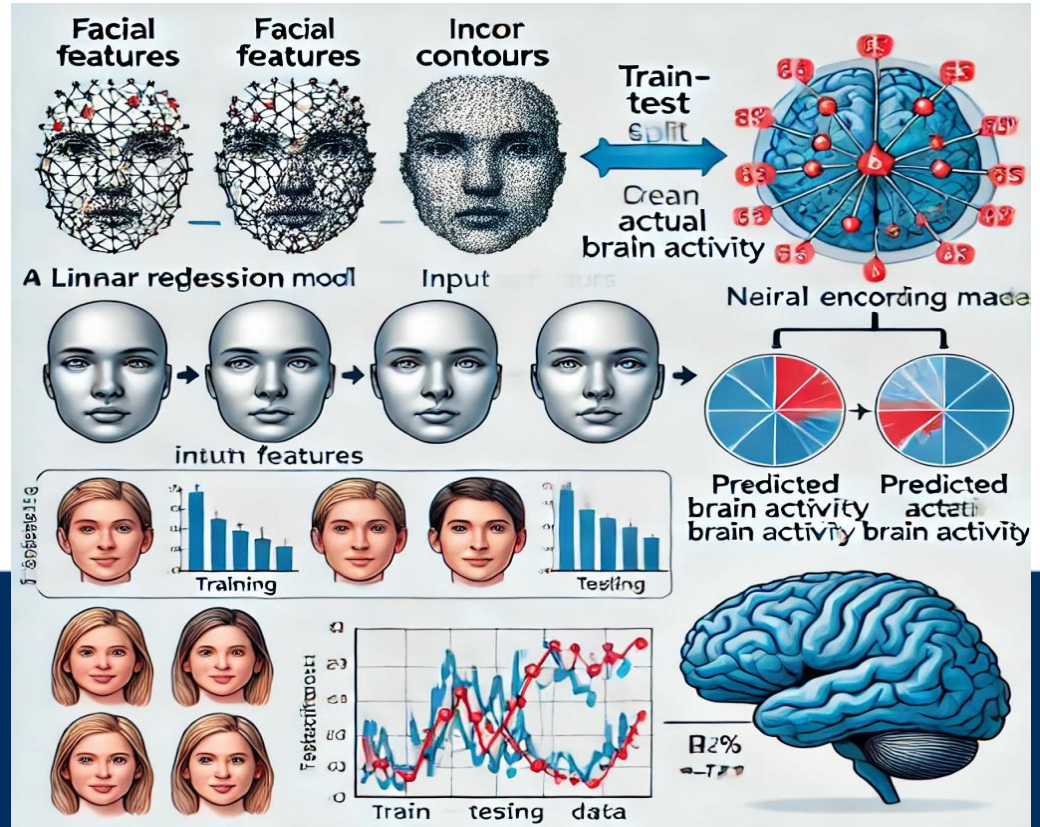


Figure 2. Face-selective activation (faces > objects, $p<0.0001$) on an inflated brain of one subject, shown from lateral and ventral views of the right and left hemispheres. Three face-selective regions are typically found: the FFA in the fusiform gyrus along the ventral part of the brain, the OFA in the lateral occipital area and the fSTS in the posterior region of the superior temporal sulcus.

3. Neural Encoding Model

- Feature extraction
- Development of regression model
- Model fitting
- Prediction of new instances



4. Model Generalization

Theoretical Ideas

Concrete Testable Model

The Hodgkin – Huxley Model

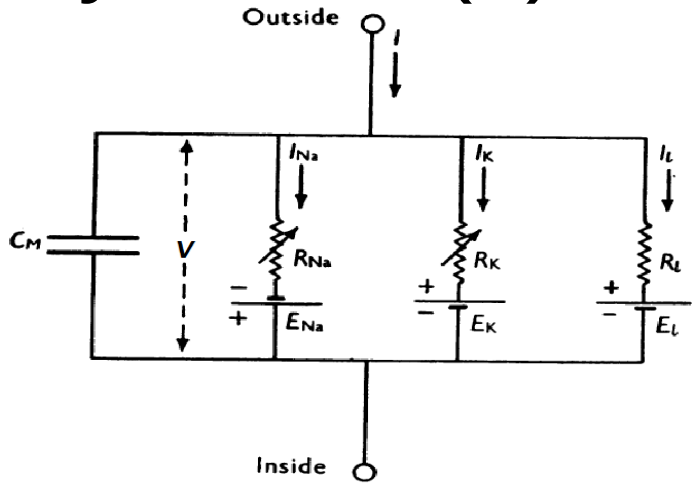
- The model developed by Alan Hodgkin and Andrew Huxley in 1952.
- It is a mathematical model that describes how action potentials (electrical signals) are initiated and propagated in neurons.
- It is one of the foundational models in computational neuroscience.
- It is based on experimental data from the giant squid axon.

- The Hodgkin - Huxley model explains how the membrane potential of a neuron changes in response to the flow of ions through voltage-gated ion channels.
- The model includes a detailed description of how sodium (Na^+) and potassium (K^+) ions move through the membrane during the different phases of an action potential.

It **accounts** for the **dynamic changes** in ions conductances and **explains** the **depolarization**, **repolarization** and **refractory** periods of a neuron's **action potential.**

Key Concepts in the Hodgkin – Huxley Model (1)

- The membrane potential V(t) is the electrical potential difference between the inside and outside of a neuron.
- In the Hodgkin – Huxley model, the change in membrane potential is due to the flow of ionic currents.
- The total current flowing across the membrane consists of:
  - Sodium current I_{Na}
  - Potassium current I_K
  - Leak current I_{leak}
- These currents are governed by the conductance of the corresponding ion channels and the difference between the membrane potential V and the equilibrium potential for each ion

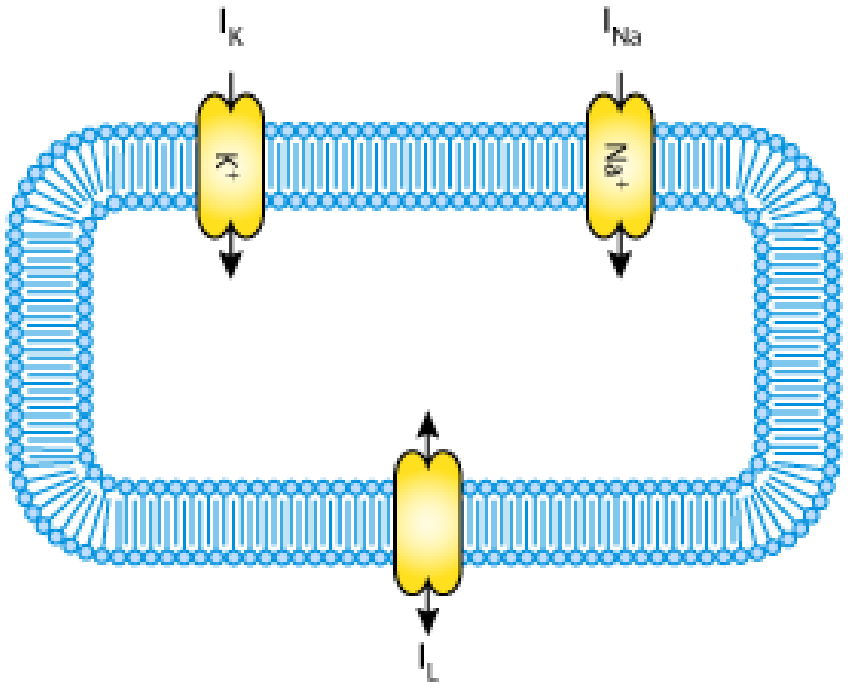


Key Concepts in the Hodgkin – Huxley Model (2)

- The Hodgkin – Huxley model is based on the principle of conservation of charge
- This means that the change in membrane potential over time is proportional to the net ionic current:

$$C_m \frac{dV(t)}{dt} = I_{\text{ext}} - (I_{\text{Na}} + I_{\text{K}} + I_{\text{leak}})$$

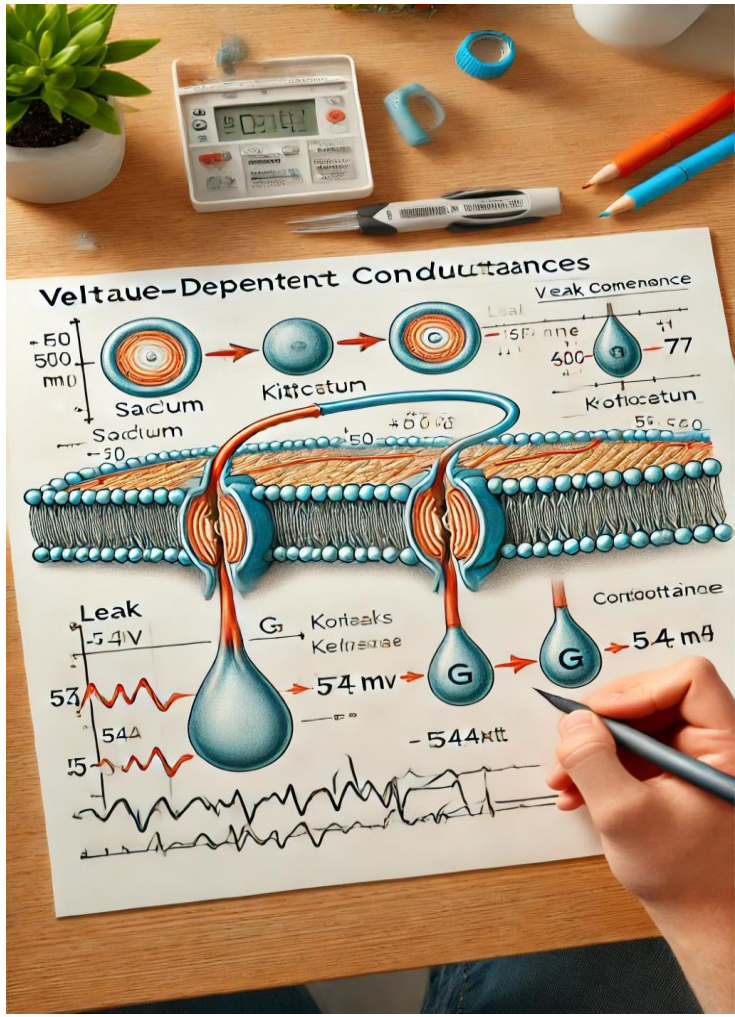
- Where, $\mathbf{C_m}$ is the membrane capacitance, representing the ability of the membrane to store charge.
- $\mathbf{I_{ext}}$ is the external current applied to the neuron (e.g. from an electrode or a synapse).
- I_{Na}, I_K and I_{leak} are the sodium, potassium and leak currents respectively.



Key Concepts in the Hodgkin – Huxley Model (3)

Ionic Conductances

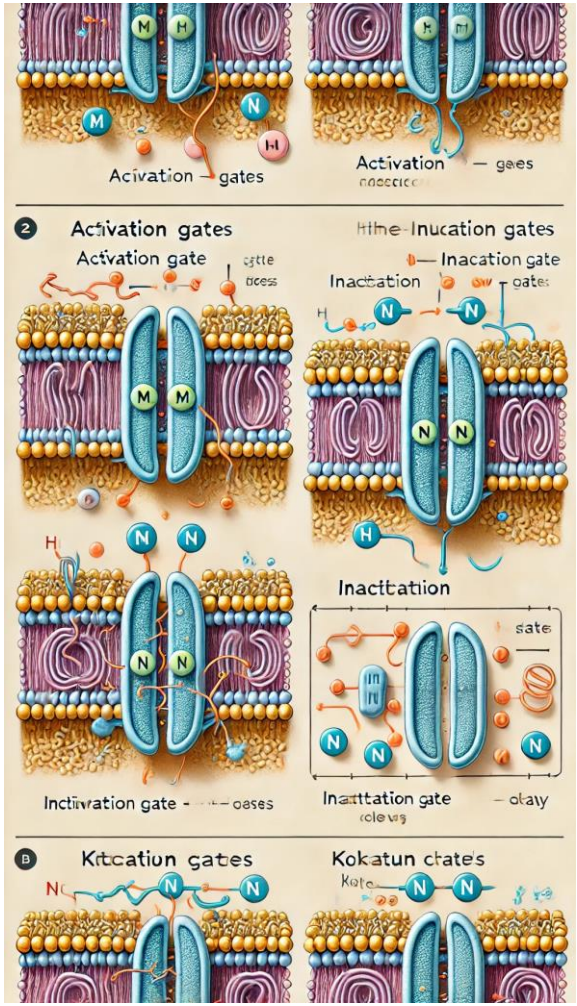
- The conductances for sodium and potassium are voltage-dependent.
- They change over time due to the opening and closing of ion channels.
- The leak conductance (g) is constant.
- The sodium reversal potential is around +50 mV.
- The potassium reversal potential is around -77 mV.
- The leak reversal potential is around -54.4 mV.



$$I_{\mathrm{Na}} = g_{\mathrm{Na}}(V)(V - E_{\mathrm{Na}}) \qquad I_{\mathrm{K}} = g_{\mathrm{K}}(V)(V - E_{\mathrm{K}}) \qquad I_{\mathrm{leak}} = g_{\mathrm{leak}}(V - E_{\mathrm{leak}})$$

Gating Variables & Ion Channels

- The conductances for sodium and potassium are controlled by gating variables.
- These represent the probability that the ion channels are open at any given time.
- Sodium channels have two types of gates (activation and inactivation):
  - The **activation gate m** controls how sodium channels open during depolarization. $g_{\mathrm{Na}}(V) = \bar{g}_{\mathrm{Na}} m^3 h$
  - The **inactivation gate h** controls how sodium channels close during the action potential.
  - The total sodium conductance is given by: , where $\overline{g_{Na}}$ is the maximum sodium conductance.
- Potassium channels have one type of gate (activation).
  - The **activation gate n** for potassium conductance controls how potassium channels open to repolarize the membrane.
  - The total potassium conductance is given by: $g_{\mathrm{K}}(V) = \bar{g}_{\mathrm{K}} n^4$, where $\overline{g_K}$ is the maximum potassium conductance.



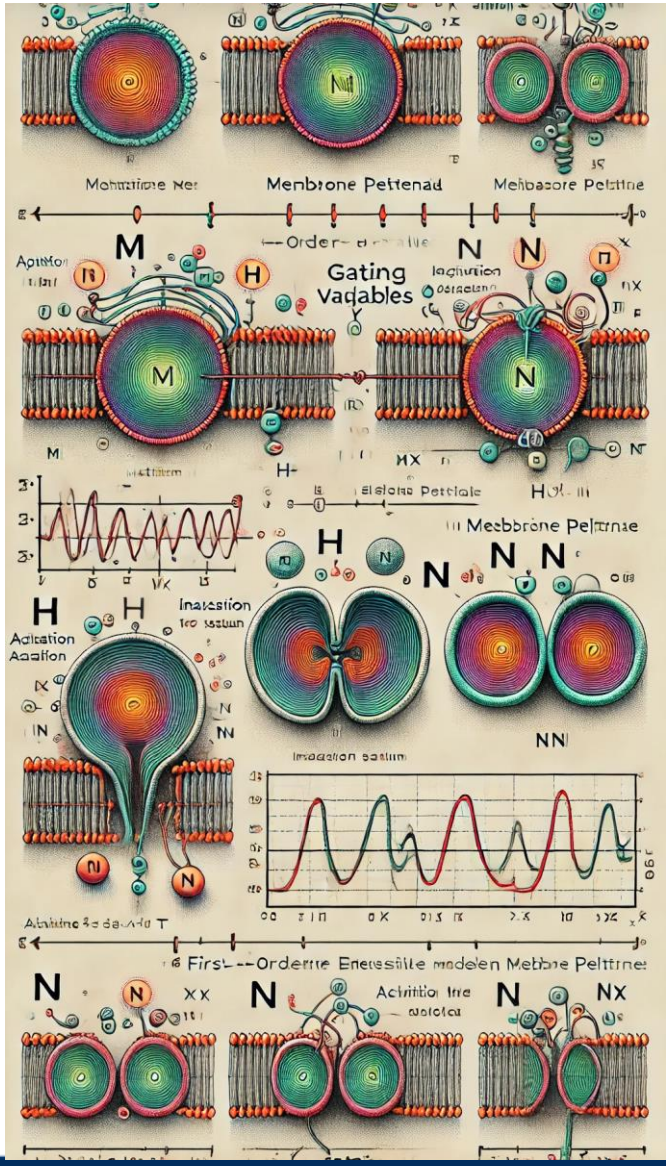
Gating Variable Dynamics

- The gating variables **m**, **h** and **n** evolve over time according to first-order differential equations.
- These equations describe how the gates open or close based on the membrane potential.
- For a generic gating variable x (which could be m, h or n):

Wher $\frac{dx}{dt} = \alpha_x(V)(1-x) - \beta_x(V)x$

- ❑ $\alpha_x(V)$ is the rate of opening of the gate
- ❑ $\beta_x(V)$ is the rate of closing of the gate
- ❑ V is the membrane potential

- The values of $\boldsymbol{\alpha_x}$ and $\boldsymbol{\beta_x}$ are voltage dependent
- They are calculated using experimentally determined functions.



Action Potential Phases

The Hodgkin – Huxley model successfully describes the different phases of an action potential:

1. At rest, the membrane potential is around **-65** mV. The **h** channel is high, **m** and **n** channels are low.
2. When the membrane potential reaches a threshold (around **-55** mV), sodium channels open rapidly (m increases), causing sodium ions to flow into the cell. This results in the **depolarization** of the membrane (the membrane potential becomes more **positive**).
3. As the membrane potential becomes more positive, sodium channels **inactivate** (**h** increases) and potassium channels **open** (**n** increases). This allows potassium ions to flow out of the cell, driving the membrane potential back down.
4. Potassium channels remain open for a short time after **repolarization**, causing the membrane potential to **overshoot** its resting value, leading to **hyperpolarization**.
5. After hyperpolarization, potassium channels **close** and the membrane potential **returns** to its resting value.

Refractory Periods

The **Hodgkin – Huxley** model also explains the **refractory periods**:

- **Absolute Refractory Period**: During this time, the sodium channels are inactivated (due to h) and the neuron cannot fire another action potential, no matter how strong the stimulus.
- **Relative Refractory Period**: After the absolute refractory period, some sodium channels have **recovered**, but the neuron requires a **stronger stimulus** to fire again because the membrane is **hyperpolarised**.

Summary of the Hodgkin – Huxley Model

- Membrane potential is determined by the balance of sodium, potassium and leak currents.
- Gating variables control the opening and closing of ion channels, and their dynamics explain the time-dependent nature of action potentials.
- The model provides a quantitative description of how action potentials are initiated, propagated and terminated in neurons.
- It explains the different phases of the action potential and the refractory periods.

➢ The Hodgkin – Huxley model remains one of the most important models in neuroscience.
➢ It is still used as a foundation for more complex models of human behaviour.

Model Parameters

- C_m: Membrane capacitance (1 uF/ cm^2) which governs how fast the membrane potential changes.
- g_{Na}, g_K and g_L: Maximum conductances for sodium, potassium and leak channels respectively.
- E_{Na}, E_K and E_L: Reversal potentials for sodium, potassium and leak channels, representing the equilibrium potentials of these ions.

```
% Hodgkin-Huxley Neuron Model Simulation

% Parameters
C_m = 1.0;     % Membrane capacitance, uF/cm^2
g_Na = 120.0; % Maximum conductance of sodium (mS/cm^2)
g_K = 36.0;    % Maximum conductance of potassium (mS/cm^2)
g_L = 0.3;     % Leak conductance (mS/cm^2)
E_Na = 50;     % Sodium reversal potential (mV)
E_K = -77;     % Potassium reversal potential (mV)
E_L = -54.4;   % Leak reversal potential (mV)
V_rest = -65; % Resting membrane potential (mV)
```

The simulation runs for **100 ms** with a time step of **0.01 ms**, creating a **time vector** from 0 to 100 ms.

```
% Time parameters
dt = 0.01;   % Time step (ms)
T = 100;     % Total time (ms)
time = 0:dt:T;  % Time vector
```

An external current of 10 microamperes is injected between 5 ms and 6 ms to trigger an action potential.

```
% External current (stimulation)
I_ext = zeros(size(time));
I_ext(500:600) = 10;  % Inject current between 5 and 6 ms (in microamps)
```

A function handle example

- The symbol '@' in Matlab defines function handles.
- It is a way to create anonymous functions, which are not stored in a separate file or given a specific name.
- '**alpha_m**' is a function that defines $\mathbf{a_m(V)}$, the rate function for the sodium activation gate m.
- '**@(V)**' is the part defining that **V** is the input argument of the function (in this case, **V** is the membrane potential).
- The expression to the right of '**@(V)**' is the function itself, which calculates the rate constant $\mathbf{a_m(V)}$ based on the value of **V**.

```
alpha_m = @(V) (0.1 * (V + 40)) / (1 - exp(-(V + 40) / 10));
```

- This means that '**alpha_m**' is a function of **V**, and when you call '**alpha_m(V)**' with a specific value of V, it will return the result of the formula: $\alpha_m(V) = \frac{0.1 \times (V + 40)}{1 - e^{-\frac{V+40}{10}}}$
- You can use '**alpha_m**' like any other function.
- For example: `result = alpha_m(-65);  % Calculate alpha_m for V = -65 mV` will calculate the value of '**alpha_m**' for V = -65.

Gating Variables

- The gating variables **m (sodium activation)**, **h (sodium inactivation)** and **n (potassium activation)** control the conductances of the sodium and potassium channels.
- The rate functions **α** and **β** govern the transition between the open and closed states of the gates and are functions of the membrane potential **V**.

- The '**alpha_m**' and '**beta_m**' are the functions describing the rates at which the **sodium activation gate m** opens and closes respectively.
- The '**alpha_h**' and '**beta_h**' are the functions describing the rates at which the **sodium inactivation gate h** opens and closes respectively.
- The '**alpha_n**' and '**beta_n**' are the functions describing the rates at which the **potassium activation gate n** opens and closes respectively.

```
% Storage for gating variables over time
m_values = zeros(size(time));
h_values = zeros(size(time));
n_values = zeros(size(time));
V_values = zeros(size(time));


% Define rate functions for gating variables
alpha_m = @(V) (0.1 * (V + 40)) / (1 - exp(-(V + 40) / 10));
beta_m = @(V) 4.0 * exp(-(V + 65) / 18);
alpha_h = @(V) 0.07 * exp(-(V + 65) / 20);
beta_h = @(V) 1.0 / (1 + exp(-(V + 35) / 10));
alpha_n = @(V) (0.01 * (V + 55)) / (1 - exp(-(V + 55) / 10));
beta_n = @(V) 0.125 * exp(-(V + 65) / 80);
```

Simulation Loop

- For each time step, the gating variables are updated using the **Euler method**.
- These variables, in turn, determine the conductances of sodium and potassium channels.
- The ionic currents I_{Na}, I_K and I_{leak} are calculated based on the conductances and the membrane potential.
- The **membrane potential V(t)** is the updated by balancing the external current and the ionic currents.

```
% Simulation loop
for t = 2:length(time)
    % Update gating variables using Euler method
    m = m + dt * (alpha_m(V(t-1)) * (1 - m) - beta_m(V(t-1)) * m);
    h = h + dt * (alpha_h(V(t-1)) * (1 - h) - beta_h(V(t-1)) * h);
    n = n + dt * (alpha_n(V(t-1)) * (1 - n) - beta_n(V(t-1)) * n);

    % Compute conductances for sodium and potassium
    g_Na_t = g_Na * (m^3) * h;
    g_K_t = g_K * (n^4);

    % Compute ionic currents
    I_Na = g_Na_t * (V(t-1) - E_Na);
    I_K = g_K_t * (V(t-1) - E_K);
    I_L = g_L * (V(t-1) - E_L);

    % Update membrane potential using Euler's method
    V(t) = V(t-1) + dt * (I_ext(t) - (I_Na + I_K + I_L)) / C_m;

    % Store values for plotting
    m_values(t) = m;
    h_values(t) = h;
    n_values(t) = n;
    V_values(t) = V(t);
end
```

Plotting

- The **membrane potential V(t)** is plotted over time to show the action potential.
- The **gating variables m**, **h** and **n** are also plotted to show how the ion channels open and close during the action potential.

```
% Plot membrane potential over time
figure;
subplot(2,1,1);
plot(time, V_values, 'LineWidth', 2);
title('Hodgkin-Huxley Model: Membrane Potential');
xlabel('Time (ms)');
ylabel('Membrane Potential (mV)');
grid on;

% Plot gating variables over time
subplot(2,1,2);
plot(time, m_values, 'r', 'LineWidth', 1.5); hold on;
plot(time, h_values, 'g', 'LineWidth', 1.5);
plot(time, n_values, 'b', 'LineWidth', 1.5);
legend('m (Na+ activation)', 'h (Na+ inactivation)', 'n (K+ activation)');
title('Gating Variables Over Time');
xlabel('Time (ms)');
ylabel('Gating Variable');
grid on;
```

What you should see

- The **membrane potential** plot should show the classic action potential waveform, with a sharp depolarization followed by repolarization and a brief hyperpolarization.
- The **gating variables m**, **h** and **n** will show how:
  - **m (sodium activation)** rises during depolarization and then falls during repolarization.
  - h (sodium inactivation) falls during depolarization and rises during repolarization.
  - n (potassium activation) rises during depolarization and remains high during repolarization, helping to bring the membrane potential back down.