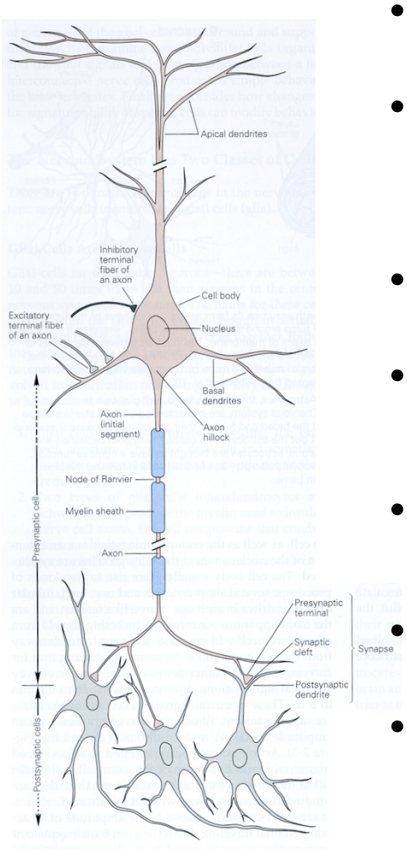
4 Concept neurons — Introduction to artificial neural networks

from Kandel, Schwartz and Jessel, *Principles of Neural Science*

4.1 Typical neuron/nerve cell:



- The **cell body** or **soma** contains the nucleus, the storehouse of genetic information.
- **Axons**, the output/transmitting element of neurons, can vary greatly in length; some can extend more than 3m within the body. Most axons in the central nervous system are very thin ($0.2 \ldots 20$ μm in diameter) compared with the diameter of the cell body (50 μm or more).
- Many axons are insulated by a **fatty sheath of myelin** that is interrupted at regular intervals by the **nodes of Ranvier**.
- The **action potential** is initiated either at the **axon hillock**, the initial segment of the axon, or in some cases slightly farther down the axon at the first nod of Ranvier.
- Branches of the axon of one neuron (the presynaptic neuron) transmit signals to another neuron (the postsynaptic cell) at a site called the **synapse**.
- The branches of a single axon may form synapses with as many as 1000 other neurons.
- The **dendrites** (apical and basal) are input elements of the neuron. Together with the cell body, they receive synaptic signals from other neurons.

Simplified functions of these very complex in their nature "building blocks" of a neuron are as follow:

- The synapses are elementary signal processing devices.
  - A synapse is a biochemical device which converts a pre-synaptic electrical signal into a chemical signal and then back into a post-synaptic electrical signal.
  - The input pulse train has its amplitude modified by parameters stored in the synapse. The nature of this modification depends on the type of the synapse, which can be either inhibitory or excitatory.
- The postsynaptic signals are aggregated and transferred along the dendrites to the nerve cell body.
- The cell body generates the output neuronal signal, activation potential, which is transferred along the axon to the synaptic terminals of other neurons.
- The frequency of firing of a neuron is proportional to the total synaptic activities and is controlled by the synaptic parameters (weights).
- The pyramidal cell can receive 10^4 synaptic inputs and it can fan-out the output signal to thousands of target cells — the connectivity difficult to achieve in the artificial neural networks.

Other examples of neurons can be found in:
`http://www.csse.monash.edu.au/courseware/cse2330/Lnts/neurons/`

4.2 A simplistic model of a biological neuron

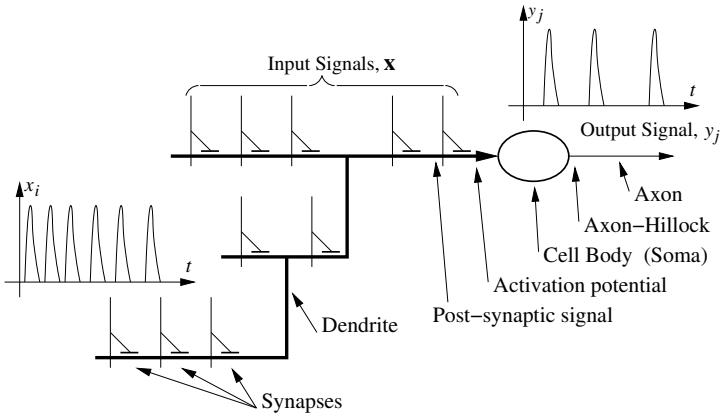


Figure 4–1: Conceptual structure of a biological neuron

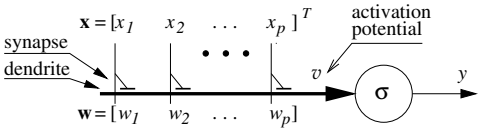
Basic characteristics of a biological neuron:

- data is coded in a form of instantaneous frequency of pulses
- synapses are either excitatory or inhibitory
- Signals are aggregated ("summed") when travel along dendritic trees
- The cell body (neuron output) generates the output pulse train of an average frequency proportional to the total (aggregated) post-synaptic activity (activation potential).

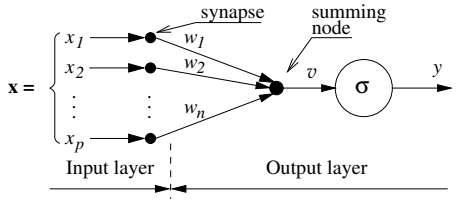
4.3 Models of artificial neurons

Three basic graphical representations of a single p-input (p-synapse) neuron:

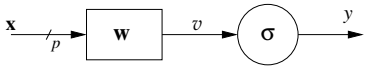
a. **Dendritic representation**



b. **Signal flow representation**



c. **Block−diagram representation**



$$v = w_1 \cdot x_1 + \cdots + w_p \cdot x_p = \mathbf{w} \cdot \mathbf{x}$$

$$y = \sigma(v)$$

Artificial neural networks are nonlinear information (signal) processing devices which are built from interconnected elementary processing devices called neurons.

An artificial neuron is a p-input single-output signal processing element which can be thought of as a simple model of a non-branching biological neuron.

From a **dendritic representation** of a single **neuron** we can identify p **synapses** arranged along a linear **dendrite** which aggregates the synaptic activities, and a neuron body or axon-hillock generating an output signal.

The **pre-synaptic activities** are represented by a p-element **column vector** of input (afferent) signals

$$\mathbf{x} = [x_1 \ \dots \ x_p]^T$$

In other words the space of input patterns is p-dimensional.

Synapses are characterised by adjustable parameters called weights or synaptic strength parameters. The weights are arranged in a p-element **row vector**:

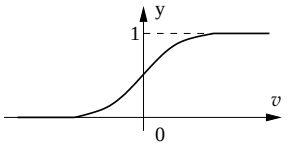
$$\mathbf{w} = [w_1 \ \dots \ w_p]$$

- In a **signal flow** representation of a neuron p synapses are arranged in a layer of input **nodes**. A dendrite is replaced by a single summing node. Weights are now attributed to **branches** (connections) between input nodes and the summing node.
- Passing through synapses and a dendrite (or a summing node), input signals are aggregated (combined) into the **activation potential**, which describes the total **post-synaptic activity**.
- The activation potential is formed as a linear combination of input signals and synaptic strength parameters, that is, as an **inner product** of the weight and input vectors:

$$v = \sum_{i=1}^{p} w_i x_i = \mathbf{w} \cdot \mathbf{x} = \begin{bmatrix} w_1 & w_2 & \cdots & w_p \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix} \tag{4.1}$$

- Subsequently, the activation potential (the total post-synaptic activity) is passed through an **activation function**, $\sigma(\cdot)$, which generates the output (efferent) signal:

$$y = \sigma(v) \tag{4.2}$$



- The activation function is typically a saturating function which normalises the total post-synaptic activity to the standard values of output (axonal) signal.

$$y = \frac{1}{1+e^{-v}} = \frac{1}{2}(\tanh(\frac{v}{2}) - 1)$$

- The **block-diagram** representation encapsulates basic operations of an artificial neuron, namely, aggregation of pre-synaptic activities, eqn (4.1), and generation of the output signal, eqn (4.2)

4.3.1 The NAND gate

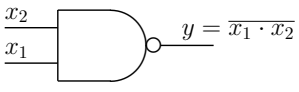
A single neuron can be used to implement some of the logic gates. Consider the NAND gate which is described by the following (truth) table:

The NAND gate

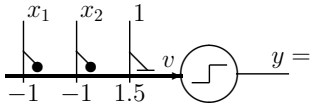
The truth table:

x_1	0	0	1	1
x_2	0	1	0	1
y	1	1	1	0

The logic symbol:



Such a gate can be implemented by the following single neuron:

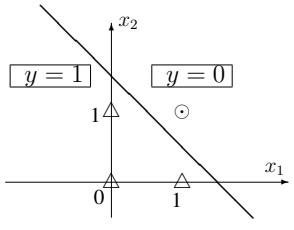


$$v = -x_1 - x_2 + 1.5$$

Note that v is as follows

x_1	0	0	1	1
x_2	0	1	0	1
v	+1.5	+0.5	+0.5	-0.5
y	1	1	1	0

The single neuron is equivalent to a straight line in a (x_1, x_2) plane.
The plane separate values of input signals for which v is either positive or negative



4.3.2 Electronics of the NAND gate

Let us consider a bit more complicated 3-input NAND gate described by the following logic equation:

$$y = \overline{a \cdot b \cdot c}$$

This equation is equivalent to a schematic as in Figure 4–2

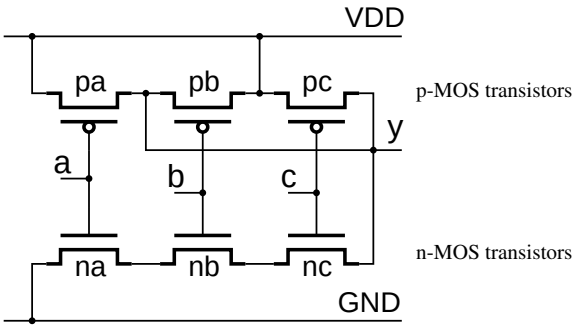


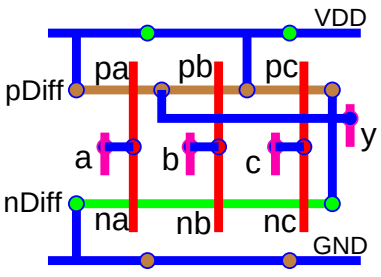
Figure 4–2: Schematic of a 3-input NAND gate

From Figure 4–2 note

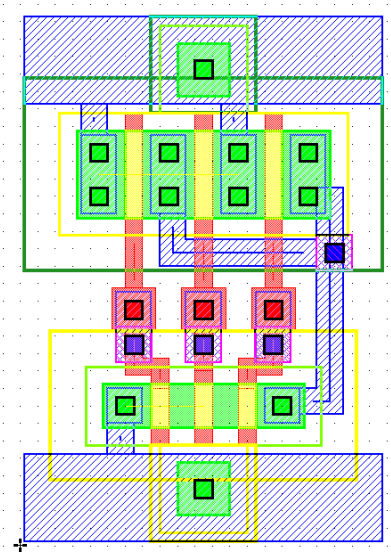
- three pMOS transistors connected in parallel between the V_{DD} and the y nodes
- three nMOS transistors connected in series between the GND and the y nodes.

The schematic can be easily converted into an equivalent "stick diagram" and finally into the circuit layout describing the plan of the circuit on the silicon surface.

The stick diagram:



The circuit layout in the AMI0.5 technology:

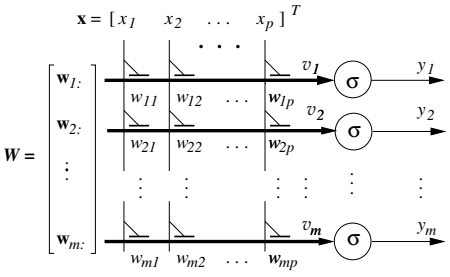


Integrated circuits are build from layers of semiconductors (doped silicon):
p-diffusion (brown in the stick diagram), n-diffusion (green),
polysilicon (red) and metal (blue, purple)

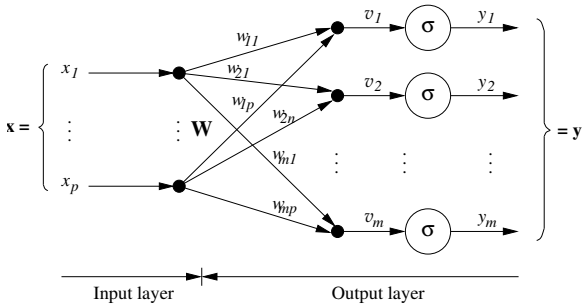
4.4 A layer of neurons

- Neurons as in sec. 4.3 can be arrange into a **layer** of neurons.
- A **single layer neural network** consists of m neurons each with the same p input signals.
- Similarly to a single neuron, the neural network can be represented in all **three** basic forms: **dendritic**, **signal-flow**, and **block-diagram** form
- From the **dendritic representation** of the neural network it is readily seen that a layer of neurons is described by a $m \times p$ matrix W of synaptic weights.
- Each row of the **weight matrix** is associated with one neuron.

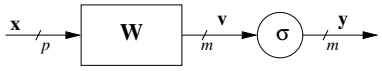
a. Dendritic graph



b. Signal–flow graph



c. Block–diagram



Operations performed by the network can be described as follows:

$$\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{1p} \\ w_{21} & \cdots & w_{2p} \\ \vdots & \cdots & \vdots \\ w_{m1} & \cdots & w_{mp} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_p \end{bmatrix} ; \quad \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} \sigma(v_1) \\ \sigma(v_2) \\ \vdots \\ \sigma(v_m) \end{bmatrix}$$

$\mathbf{v} = W \cdot \mathbf{x}$; $\mathbf{y} = \boldsymbol{\sigma}(W \cdot \mathbf{x}) = \boldsymbol{\sigma}(\mathbf{v})$; $\mathbf{v}$ is a vector of activation potentials.

- From the **signal-flow graph** it is visible that each weight parameter w_{ij} (synaptic strength) is now related to a connection between nodes of the input layer and the output layer.
- Therefore, the name **connection strengths** for the weights is also justifiable.
- The **block-diagram** representation of the single layer neural network is the most compact one, hence often most convenient to use.
- A **layer of neurons** can be connected together to form a multi-layer structure.
- Multi-layer feedforward networks are known also as **Multilayer Perceptrons** (MLPs).

4.5 Static and Dynamic Systems — General Concepts

Static systems — feedforward networks

Neural networks considered in previous sections belong to the class of **static** systems which can be fully described by a set of m-functions of p-variables as in Figure 4–3.

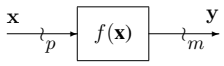


Figure 4–3: A static system: $\mathbf{y} = f(\mathbf{x})$

The defining feature of the static systems is that they are **time-independent** — current outputs depends only on the current inputs in the way specified by the mapping function, f.
Such a function can be very complex.

Dynamic systems — Recurrent Neural Networks

In the dynamic systems, the current output signals depend, in general, on current and past input signals.

There are two equivalent classes of dynamic systems: continuous-time and discrete-time systems.

The dynamic neural networks are referred to as **recurrent neural networks**.

4.6 Continuous-time dynamic systems

- Continuous-time dynamic systems operate with signals which are functions of a continuous variable, t, interpreted typically as **time**. A **spatial variable** can be also used.
- Continuous-time dynamic systems are described by means of **differential equations**. The most convenient yet general description uses only **first-order** differential equations in the following form:

$$\dot{\mathbf{y}}(t) = f(\mathbf{x}(t), \mathbf{y}(t)) \tag{4.3}$$

  where

$$\dot{\mathbf{y}}(t) \stackrel{\text{df}}{=} \frac{d\mathbf{y}(t)}{dt}$$

  is a vector of time derivatives of output signals.
- In order to model a dynamic system, or to obtain the output signals, the **integration** operation is required. The dynamic system of eqn (4.3) is illustrated in Figure 4–4.

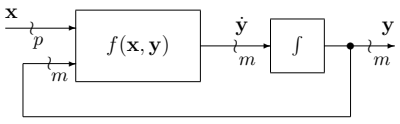


Figure 4–4: A continuous-time dynamic system: $\dot{\mathbf{y}}(t) = f(\mathbf{x}(t), \mathbf{y}(t))$

- It is evident that **feedback** is inherent to dynamic systems.

4.6.1 Discrete-time dynamic systems

- Discrete-time dynamic systems operate with signals which are functions of a discrete variable, n, interpreted typically as time, but a discrete spatial variable can be also used.
- Typically, the discrete variable can be thought of as a **sampled** version of a continuous variable:

$$t = n \cdot t_s \; ; \quad t \in \mathcal{R} \; , \quad n \in \mathcal{N}$$

  and t_s is the **sampling time**
- Analogously, discrete-time dynamic systems are described by means of **difference equations**.
- The most convenient yet general description uses only **first-order** difference equations in the following form:

$$\mathbf{y}(n+1) = f(\mathbf{x}(n), \mathbf{y}(n)) \tag{4.4}$$

  where $\mathbf{y}(n+1)$ and $\mathbf{y}(n)$ are the predicted (future) value and the current value of the vector $\mathbf{y}$, respectively.

- In order to model a discrete-time dynamic system, or to obtain the output signals, we use the **unit delay** operator, $D = z^{-1}$ which originates from the z-transform used to obtain analytical solutions to the difference equations.
- Using the delay operator, we can re-write the first order difference equation into the following operator form:

$$z^{-1}\mathbf{y}(n+1) = \mathbf{y}(n)$$

  which leads to the structure as in Figure 4–5.

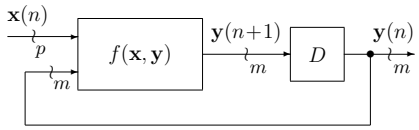


Figure 4–5: A discrete-time dynamic system: **y**(n+1) = f(**x**(n), **y**(n))

- Notice that **feedback** is also present in the discrete dynamic systems.

4.6.2 Example: A continuous-time generator of a sinusoid

- As a simple example of a continuous-time dynamic system let us consider a linear system which generates a sinusoidal signal.
- Eqn (4.3) takes on the following form: $$\dot{\mathbf{y}}(t) = A \cdot \mathbf{y}(t) + B \cdot \mathbf{x}(t) \tag{4.5}$$

  where $$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \; ; \quad A = \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix} \; ; \quad B = \begin{bmatrix} 0 \\ b \end{bmatrix} \; ; \quad \mathbf{x} = \delta(t)$$

  $\delta(t)$ is the unit impulse which is non-zero only for $t = 0$ and is used to describe the initial condition.
- In order to show that eqn (4.5) really describes the sinusoidal generator we re-write this equation for individual components. This yields:
$$\begin{aligned} \dot{y}_1 &= \omega\, y_2 \\ \dot{y}_2 &= -\omega\, y_1 + b\, \delta(t) \end{aligned} \tag{4.6}$$
- Differentiation of the first equation and substitution of the second one gives the second-order linear differential equation for the output signal y_1:
$$\ddot{y}_1 + \omega^2 y_1 = \omega\, b\, \delta(t)$$
- Taking the Laplace transform and remembering that $\mathcal{L}\delta(t) = 1$, we have:
$$y_1(s) = b\frac{\omega}{s^2 + \omega^2}$$

- Taking the inverse Laplace transform we finally have
$$y_1(t) = b \sin(\omega t)$$
- The internal structure of the generator can be obtained from eqns (4.7) and illustrated using the dendritic representation as in Figure 4–6.

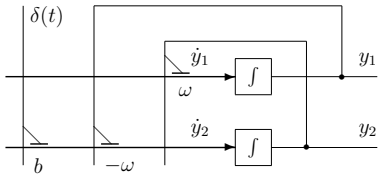


Figure 4–6: A continuous-time sinusoidal generator

- The generator can be thought of as a simple example of a **linear recurrent neural network** with the fixed weight matrix of the form:
$$W = [B\ A] = \begin{bmatrix} 0 & 0 & \omega \\ b & -\omega & 0 \end{bmatrix}$$
- The weights were designed appropriately rather than "worked out" during the learning procedure.

4.6.3 Example: A discrete-time generator of a sinusoid

- It is possible to build a discrete-time version of the sinusoidal generator using difference equations of the general form as in eqn (4.4):

$$\mathbf{y}(n+1) = A \cdot \mathbf{y}(n) + B \cdot \mathbf{x}(n) \tag{4.7}$$

where

$$A = \begin{bmatrix} \cos\Omega & \sin\Omega \\ -\sin\Omega & \cos\Omega \end{bmatrix} ; \quad B = \begin{bmatrix} 0 \\ b \end{bmatrix} ; \quad \mathbf{x}(n) = \delta(n)$$

- This time we take the z-transform directly of eqn (4.7), which gives:

$$(zI - A)\mathbf{y}(z) = B ; \quad \text{where } I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \text{and } \mathcal{Z}\delta(n) = 1$$

Hence

$$\mathbf{y}(z) = (zI - A)^{-1}B$$

and subsequently

$$\mathbf{y}(z) = (\begin{bmatrix} z - \cos\Omega & \sin\Omega \\ -\sin\Omega & z - \cos\Omega \end{bmatrix} \begin{bmatrix} 0 \\ b \end{bmatrix})/(z^2 - 2z\cos\Omega + 1)$$

Extracting the first component, we have

$$y_1(z) = \frac{b\sin\Omega}{z^2 - 2z\cos\Omega + 1}$$

Taking the inverse z-transform finally yields

$$y_1(n+1) = b\sin(\Omega n)$$

which means that the discrete-time dynamic system described by eqn (4.7) generates a sinusoidal signal.

- The structure of the generator is similar to the previous one and is presented in Figure 4–7.

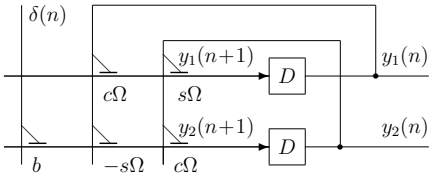


Figure 4–7: A discrete-time sinusoidal generator

- This time the weight matrix is:

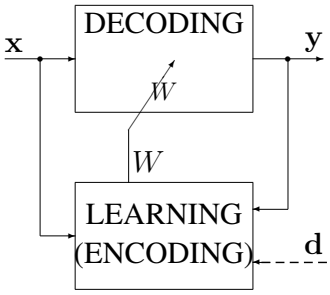
$$W = [B\ A] = \begin{bmatrix} 0 & c\Omega & s\Omega \\ b & -s\Omega & c\Omega \end{bmatrix}$$

where

$$s\Omega = \sin\Omega, \quad \text{and } c\Omega = \cos\Omega$$

4.7 Introduction to learning

- In the previous sections we concentrated on the **decoding** part of a neural network assuming that the **weight matrix**, W, is given.
- If the weight matrix is satisfactory, during the decoding process the network performs some useful task it has been design to do.
- In simple or specialised cases the weight matrix can be pre-computed, but more commonly it is obtained through the **learning** process.
- Learning is a dynamic process which modifies the weights of the network in some desirable way. As any dynamic process learning can be described either in the continuous-time or in the discrete-time framework.
- A neural network with learning has the following structure:



- The learning can be described either by differential equations (continuous-time)

$$\dot{W}(t) = L(\, W(t), \mathbf{x}(t), \mathbf{y}(t), \mathbf{d}(t)\,) \tag{4.8}$$

  or by the difference equations (discrete-time)

$$W(n+1) = L(\, W(n), \mathbf{x}(n), \mathbf{y}(n), \mathbf{d}(n)\,) \tag{4.9}$$

  where $\mathbf{d}$ is an external teaching/supervising signal used in **supervised learning**.
- This signal in not present in networks employing **unsupervised learning**.
- The discrete-time learning law is often used in a form of a **weight update** equation:

$$\begin{aligned} W(n+1) &= W(n) + \Delta W(n) \\ \Delta W(n) &= L(\, W(n), \mathbf{x}(n), \mathbf{y}(n), \mathbf{d}(n)\,) \end{aligned} \tag{4.10}$$