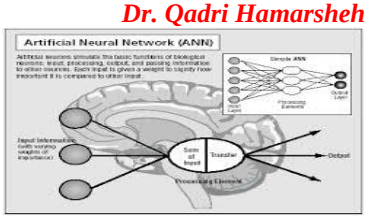
Dr. Qadri Hamarsheh



Neural Networks and Fuzzy Logic (630514)

Lecture 2

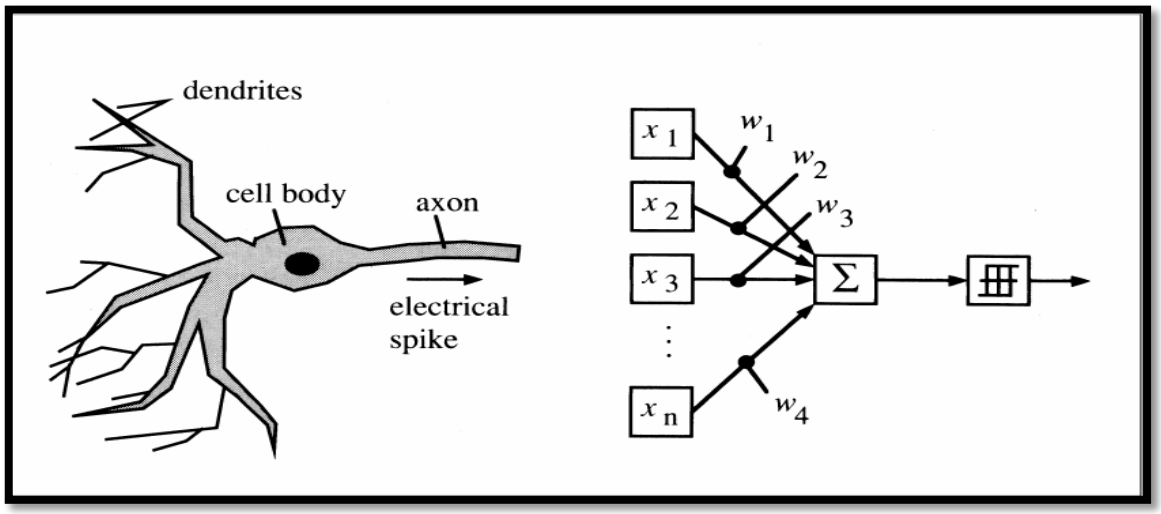
Outline

1. **Simple neuron model**
2. **Components of artificial neural networks**
3. **Common activation functions**
4. **MATLAB representation of neural network.**

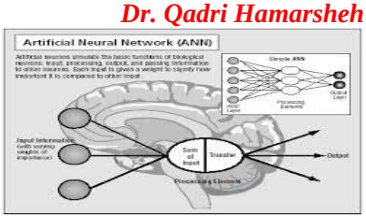
Single neuron model

Simple neuron model

Components of simple neuron



- **Input Vector**: The input of technical neurons consists of many components, therefore it is a vector.
- **Scalar output**: The output of a neuron is a **scalar**.
- **Synapses change input**: In technical neural networks the inputs are preprocessed, too. They are multiplied by a number (the weight) – they are weighted. The set of such weights represents the information storage of a neural network.
- **Accumulating the inputs**: In biology, the inputs are summarized to a pulse according to the chemical change – on the technical side this is often realized by the weighted sum.
- **Non-linear characteristic**: The input of our technical neurons is also not proportional to the output.
- **Adjustable weights**: The weights weighting the inputs are **variable**, similar to the chemical processes at the synaptic cleft. This adds a great dynamic to the network because a large part of the "**knowledge**" of a neural network is **saved in the weights**.



Simple neuron model:

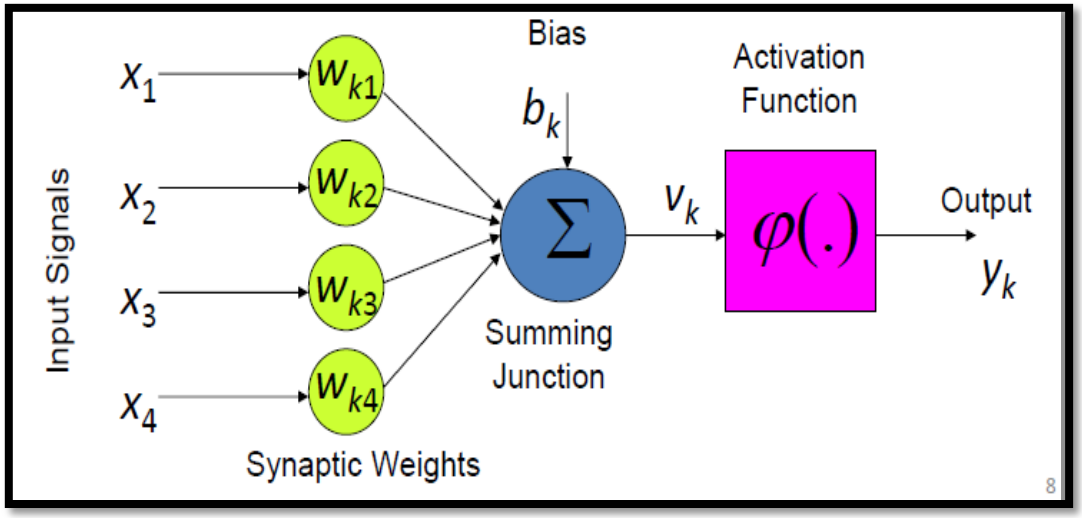
- input vector $\vec{x}$ with components x_i, These are multiplied by the appropriate weights w_i and accumulated (called **weighted sum**) :

$$\sum_i w_i x_i$$

- **nonlinear mapping** f defines the scalar output y:

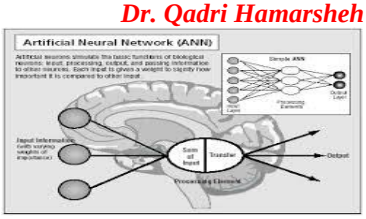
$$y = f\left(\sum_i w_i x_i\right)$$

- The model consists of a set of synapses each of which is characterized by a weight or strength of its own. An adder, an activation function and a bias.



Components of artificial neural networks

- A neural network is characterized by
  1) Its pattern of connections between the neurons (called its architecture).
  2) Its method of determining the weights on the connections (called its training, or learning, algorithm).
  3) Its activation function.
- Technical neural network consists of **simple processing units**, the neurons, and **directed, weighted connections** between those neurons.
- The reactions of the neurons to the input values depend on this **activation state**. The activation state indicates the extent of a neuron's activation and is often shortly referred to as **activation** (Neurons get activated if the network input exceeds their threshold value).
- **Threshold value**: Let **j** be a neuron. The threshold value θ_j is uniquely assigned to **j** and marks the position of the **maximum gradient** value of the activation function.



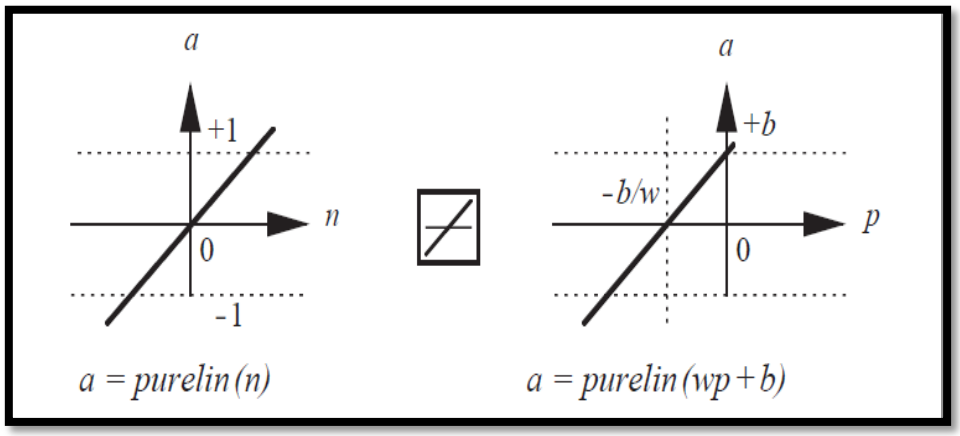
- The **bias neuron** is a technical trick to consider **threshold values as connection weights:** threshold values for neurons can also be realized as connecting weight of a continuously firing neuron.
- The activation function is also called **transfer function.**

Common activation functions

1) Identity function

$$f(x) = x \quad \text{for all } x$$

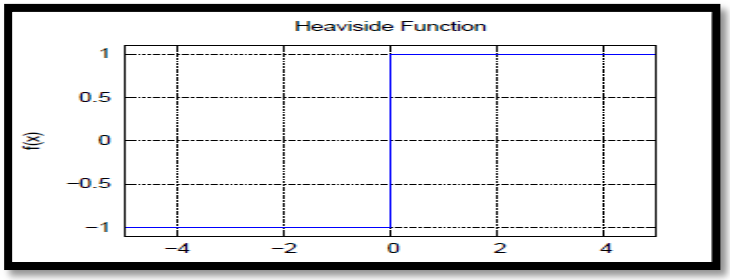
Matlab implementation (purelin)



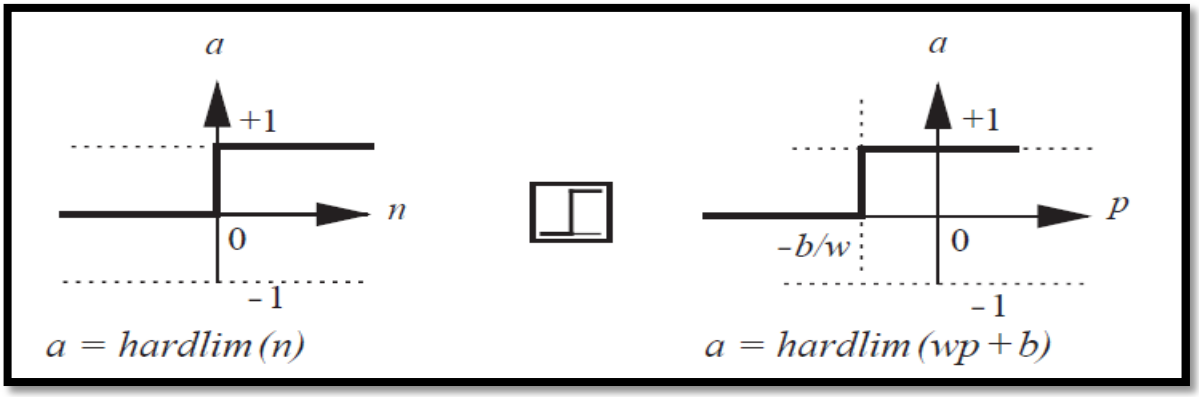
Neurons with this transfer function are used in the ADALINE networks

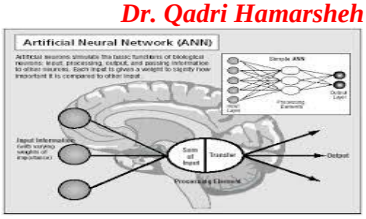
2) binary threshold function (Heaviside function)

The simplest activation function which can only take on **two values**: If the input is above a certain **threshold**, the function changes from one value to another, but otherwise remain constant. This implies that the function is **not differentiable** at the threshold and for the rest the derivative is **0**. Due to this fact, **backpropagation learning, for example, is impossible**.



Matlab implementation (Hard Limit)



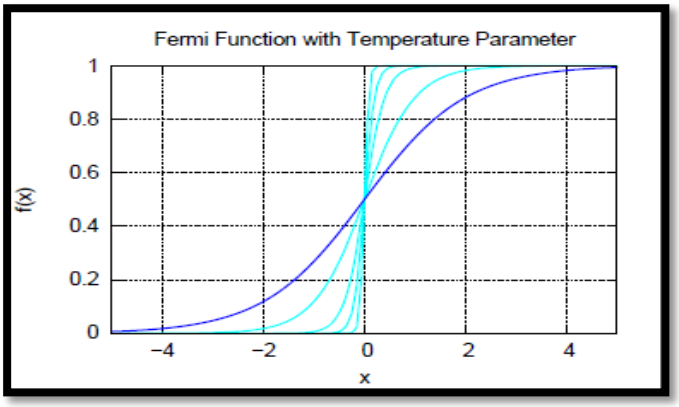


3) **Fermi function or logistic sigmoid function (an S-shaped curve)** , This transfer function takes the input (which may have any value between plus and minus infinity) and squashes the output into the range 0 to 1, according to the expression

$$\frac{1}{1+e^{-x}},$$

The Fermi function can be expanded by a **temperature parameter T** into the form:

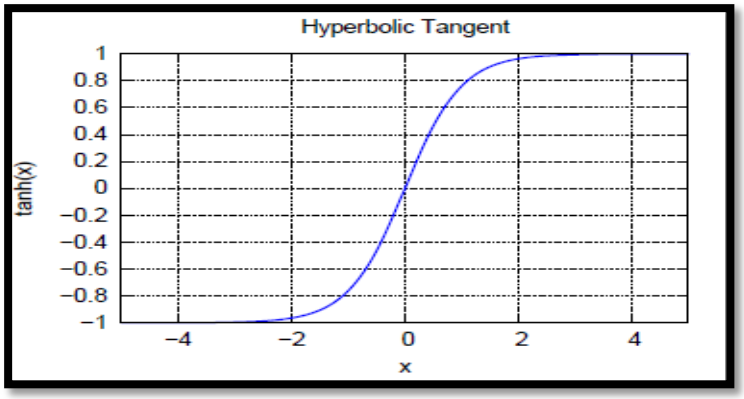
$$\frac{1}{1+e^{\frac{-x}{T}}}.$$

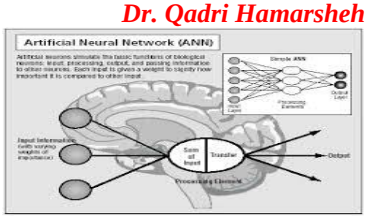


- The log-sigmoid transfer function is commonly used in multilayer networks that are trained using the backpropagation algorithm, in part because this function is **differentiable.**

4) **Hyperbolic tangent** which maps to **(−1, 1),** this function is **differentiable**.

$$\varphi(v) = \tanh\left(\frac{v}{2}\right) = \frac{1-\exp(-v)}{1+\exp(-v)}$$

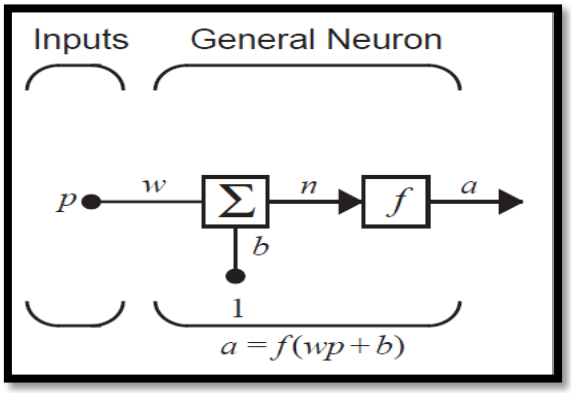




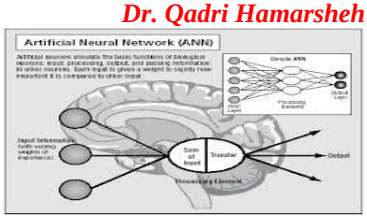
Name	Input/Output Relation	Icon	MATLAB Function
Hard Limit	$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$		hardlim
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$		hardlims
Linear	$a = n$		purelin
Saturating Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n \leq 1$ $a = 1 \quad n > 1$		satlin
Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = 1 \quad n > 1$		satlins
Log-Sigmoid	$a = \frac{1}{1+e^{-n}}$		logsig
Hyperbolic Tangent Sigmoid	$a = \frac{e^{n}-e^{-n}}{e^{n}+e^{-n}}$		tansig
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n$		poslin
Competitive	$a = 1$ neuron with max n $a = 0$ all other neurons	C	compet

MATLAB representation of neural network

Single neuron model: Single-Input Neuron



Single-Input Neuron



- The neuron output is calculated as $\mathbf{a = f(wp + b)}$,

If, for instance

$$\mathbf{w = 3,\ p = 2 \text{ and } b = -1.5,}$$

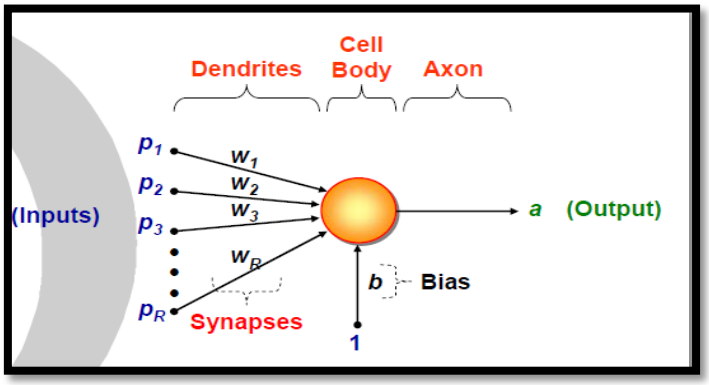
Then

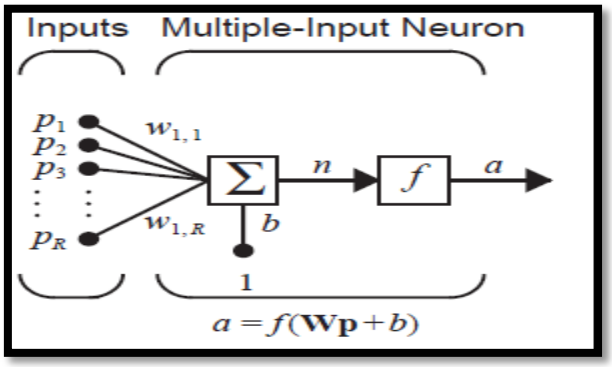
$$\mathbf{a = f(3 \cdot 2 - 1.5) = f(4.5)} \cdot$$

- The actual output depends on the particular transfer function that is chosen.

Single neuron model: Multiple-Input Neuron

- The individual inputs $\mathbf{p_1, p_2, p_3, \ldots, p_R}$ are each weighted by corresponding elements $\mathbf{w_{1,1}, w_{1,2}, w_{1,3}, \ldots, w_{1,R}}$ of the weight matrix $\mathbf{W}$





Multiple-Input Neuron

- The net input $\boldsymbol{n}$:

$$n = w_{1,1}p_1 + w_{1,2}p_2 + \cdots + w_{1,R}p_R + b .$$

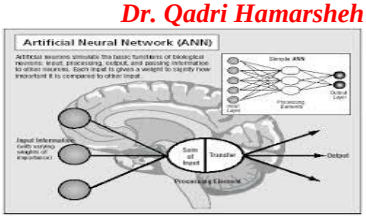
- This expression can be written in matrix form:

$$n = \mathbf{W}\mathbf{p} + b ,$$

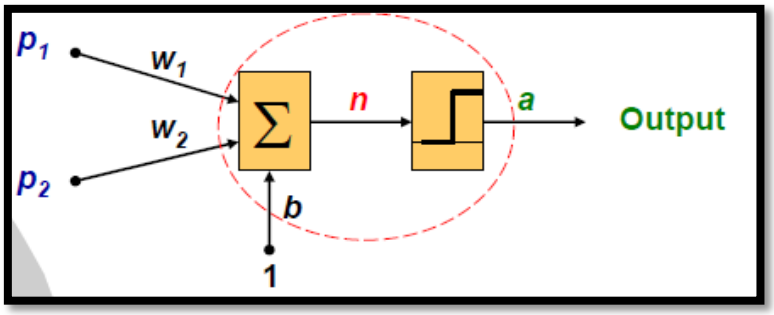
- The neuron output can be written as

$$a = f(\mathbf{W}\mathbf{p} + b) .$$

- **Weight Indices**: The first index indicates the neuron destination. The second index indicates the source of the signal fed to the neuron. Thus, the $\mathbf{w_{1,2}}$ represents the connection to the first neuron from the second source.



Example:



If *p1* = 2.5; *p2* = 3; *w1* = 0.5; *w2* = -0.7; *b* = 0.3. Let's assume the transfer function of the neuron is *hardlimit*

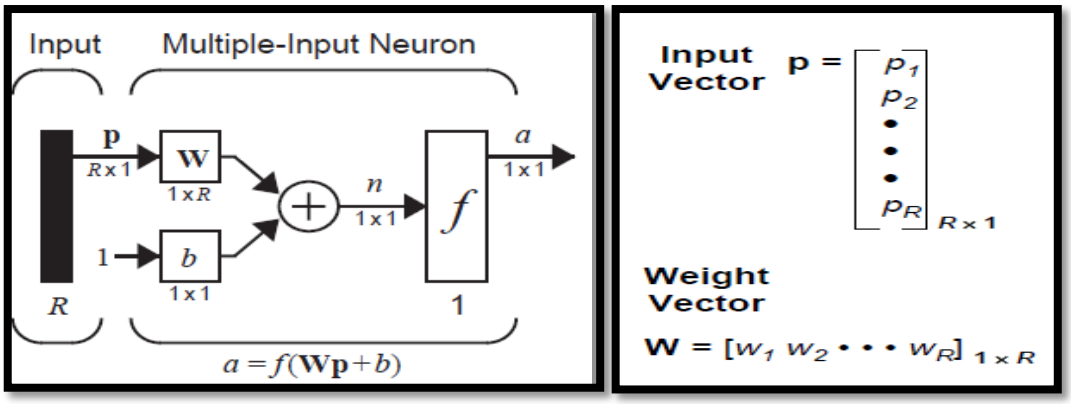
Solution

a = *hardlim*(n) = *hardlim*($w_1p_1 + w_2p_2 + b$)

= *hardlim*(0.5×2.5 + (-0.7)×3 + 0.3)

= *hardlim*(-0.55) = 0

- **Abbreviated Matlab notation**



Neuron with R Inputs, Abbreviated Notation