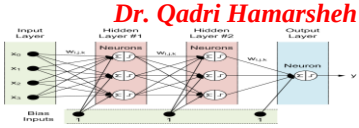
Dr. Qadri Hamarsheh



Neural Networks and Fuzzy Logic (630514)

Lecture 3

MATLAB representation of neural network

Outline

- **Neural network with single-layer of neurons.**
- **Neural network with multiple-layer of neurons.**

Introduction:

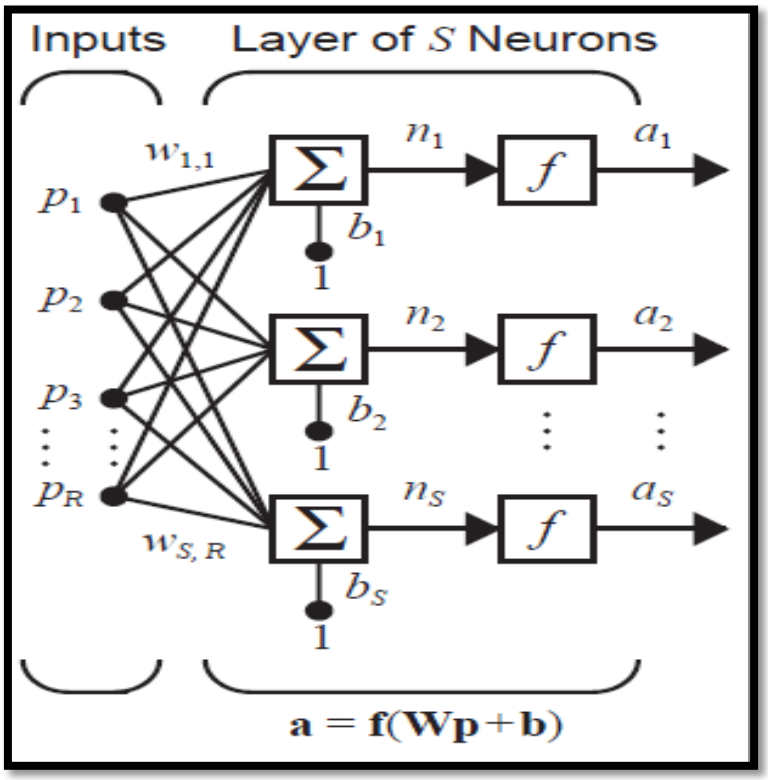
Neural Network topologies (Typical Architectures)

- **The single-layer and multilayer nets.**
- **feedforward and recurrent net:**
  - In a feedforward network each neuron in one layer has only directed connections to the neurons of the next layer (towards the output layer).
  - Recurrent networks contain feedback connections.

Feedforward Neural Network

- Feedforward contains:
  - One input layer,
  - **n** hidden processing layers (invisible from the outside, that's why the neurons are also referred to as hidden neurons)
  - One output layer.
- Every neuron is connected to all neurons of the next layer (these layers are called completely linked).

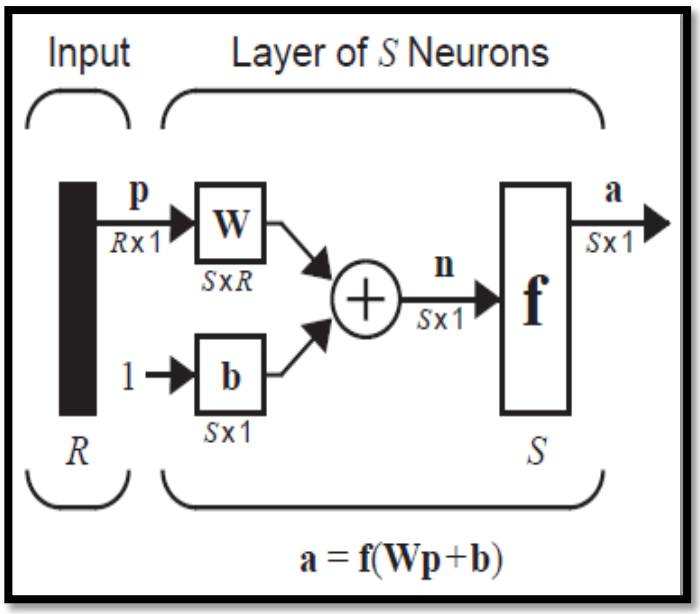
Single-Layer Net



- The units can be distinguished as input units, which receive signals from the outside world, and output units, from which the response of the net can be read. A one-layer network, with **R** input elements and **S** neurons, is shown below.

In this network:

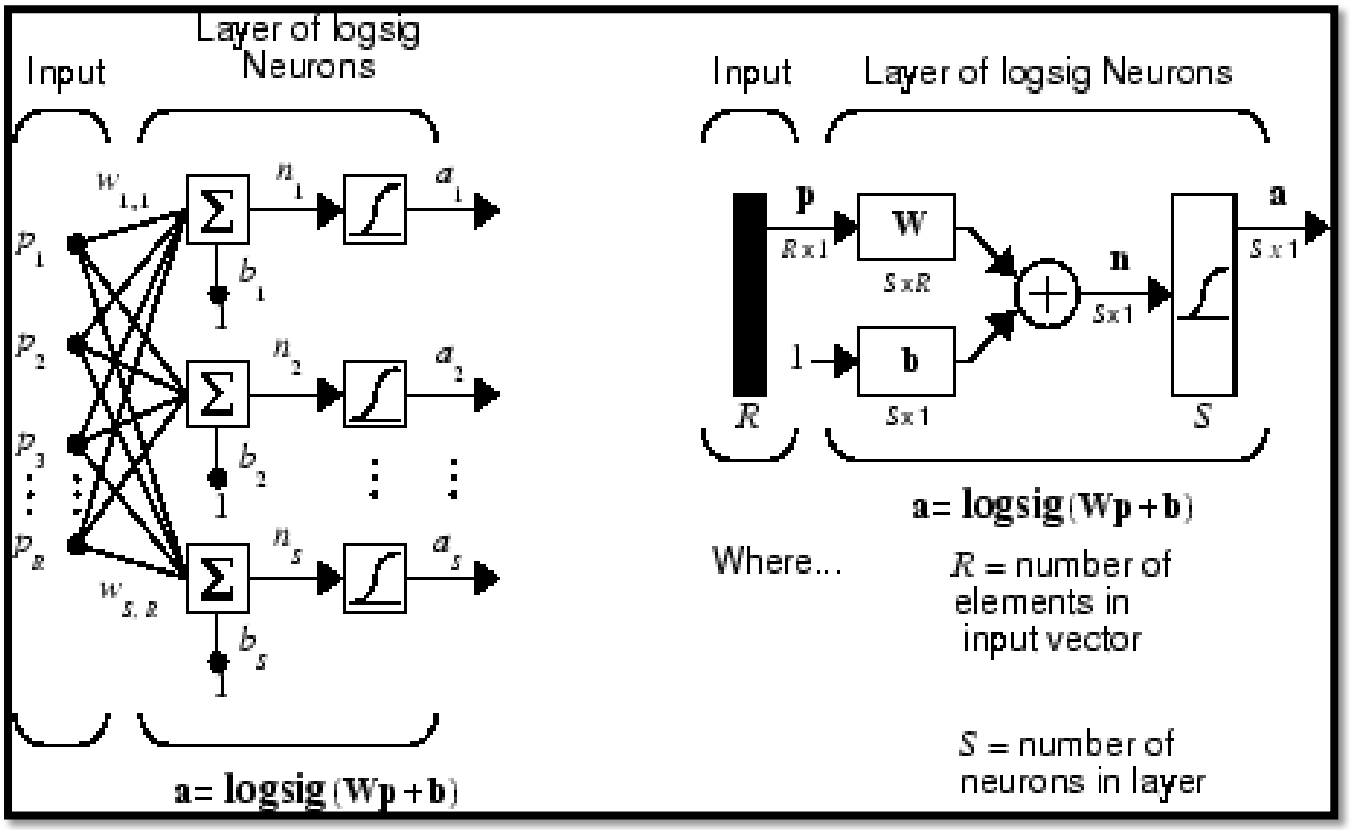
- Each element of the input vector **p** is connected to each neuron input through the weight matrix **W**.
- In MATLAB, the ***S***-neuron, ***R***-input, **one-layer** network also can be drawn in abbreviated notation

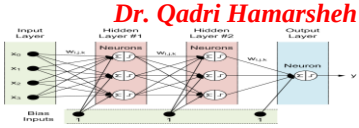


$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ \vdots & \vdots & & \vdots \\ w_{S,1} & w_{S,2} & \cdots & w_{S,R} \end{bmatrix}$$

$$\mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_R \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_S \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_S \end{bmatrix}$$

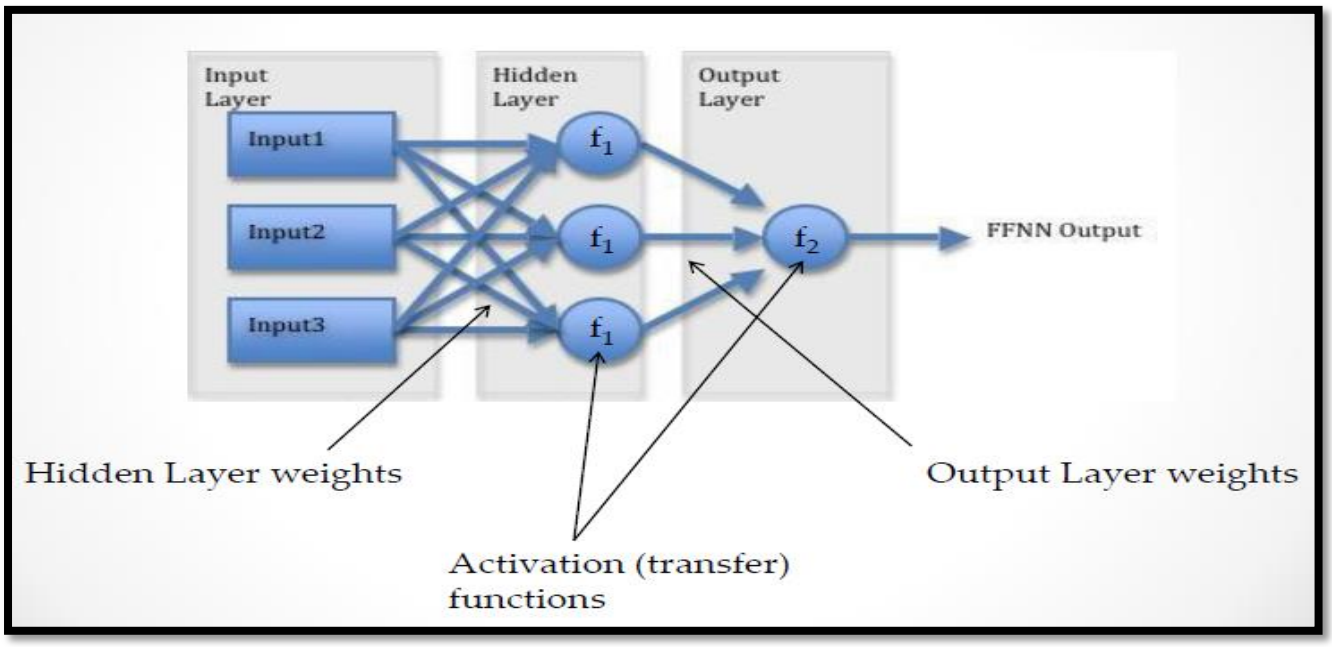
- Note that the row indices on the elements of matrix **W** indicate the destination neuron of the weight, and the column indices indicate which source is the input for that weight.
- A single-layer network of **S logsig** neurons having **R** inputs is shown below in full detail on the left and with a layer diagram on the right.





Multiple Feed-Forward Layers of Neurons

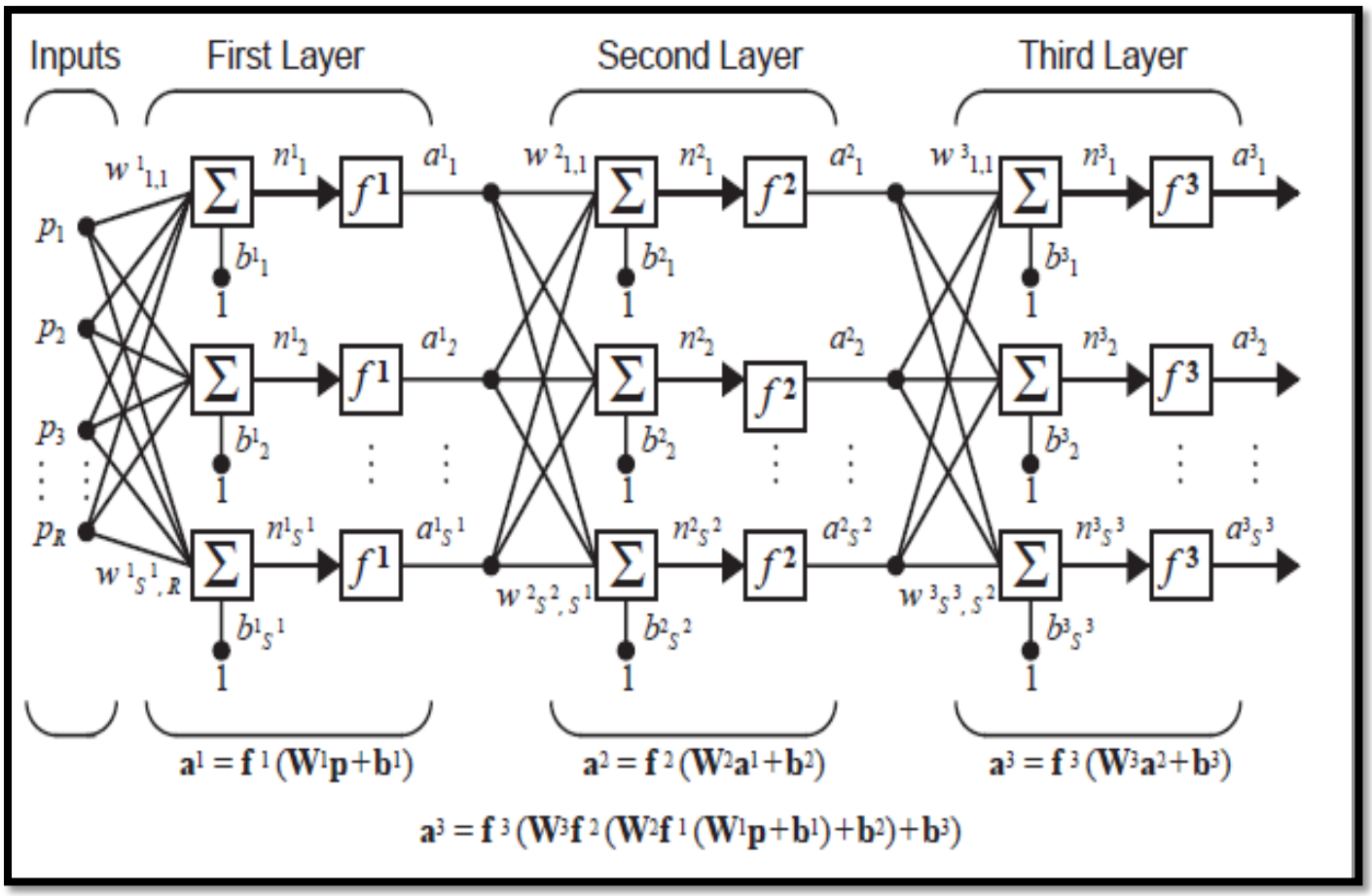
- The architecture of a multi-layer feed forward neural network is as follows:



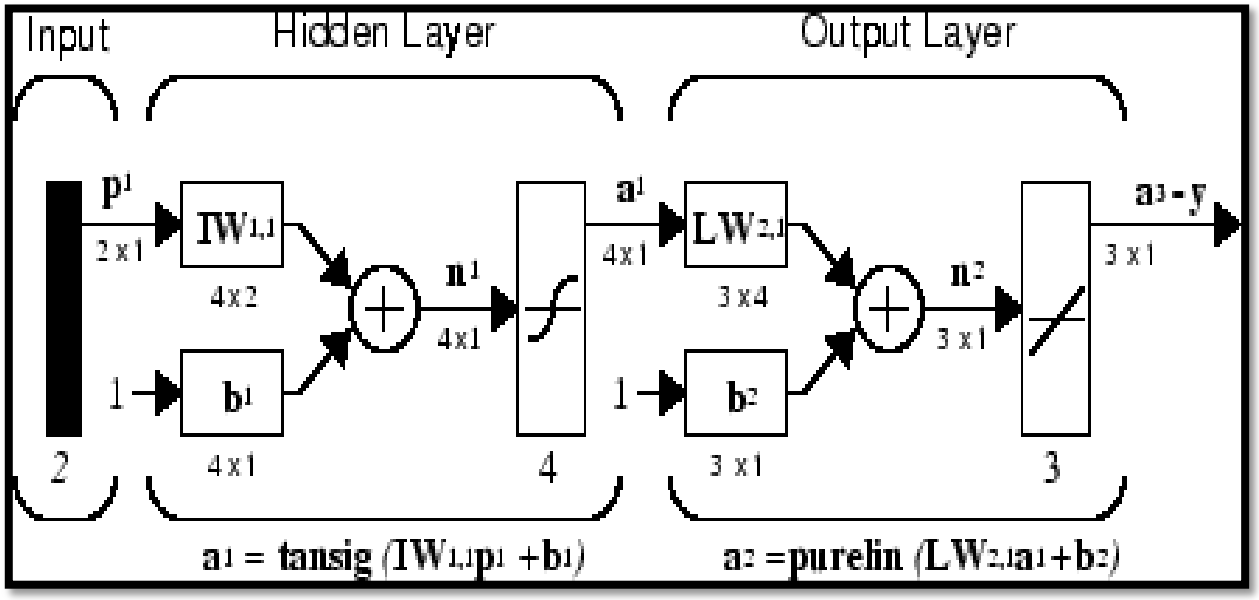
- Multilayer networks are more powerful than single-layer networks. For instance, a *two-layer network having a sigmoid first layer and a linear second layer can be trained to approximate most functions arbitrarily well. Single layer networks cannot do this*.
- The number of inputs to the network and the number of outputs from the network are defined by external problem specifications. *So if there are four external variables to be used as inputs, there are four inputs to the network. Similarly, if there are to be seven outputs from the network, there must be seven neurons in the output layer.*
- The desired characteristics of the output signal also help to select the transfer function for the output layer. *If an output is to be either 1 or -1, then a symmetrical hard limit transfer function should be used*.

Number of network inputs = number of problem inputs
Number of neurons in output layer = number of problem outputs
Output layer transfer function choice at least partly determined by problem specification of the outputs

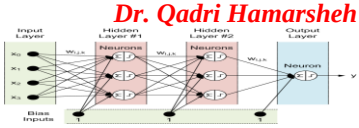
- Feedforward networks often have one or more hidden layers of sigmoid neurons followed by an output layer of linear neurons.
- Multiple layers of neurons with nonlinear transfer functions allow the network to learn nonlinear relationships between input and output vectors.
- We will call weight matrices connected to inputs, input weights; and we will call weight matrices coming from layer outputs, layer weights.



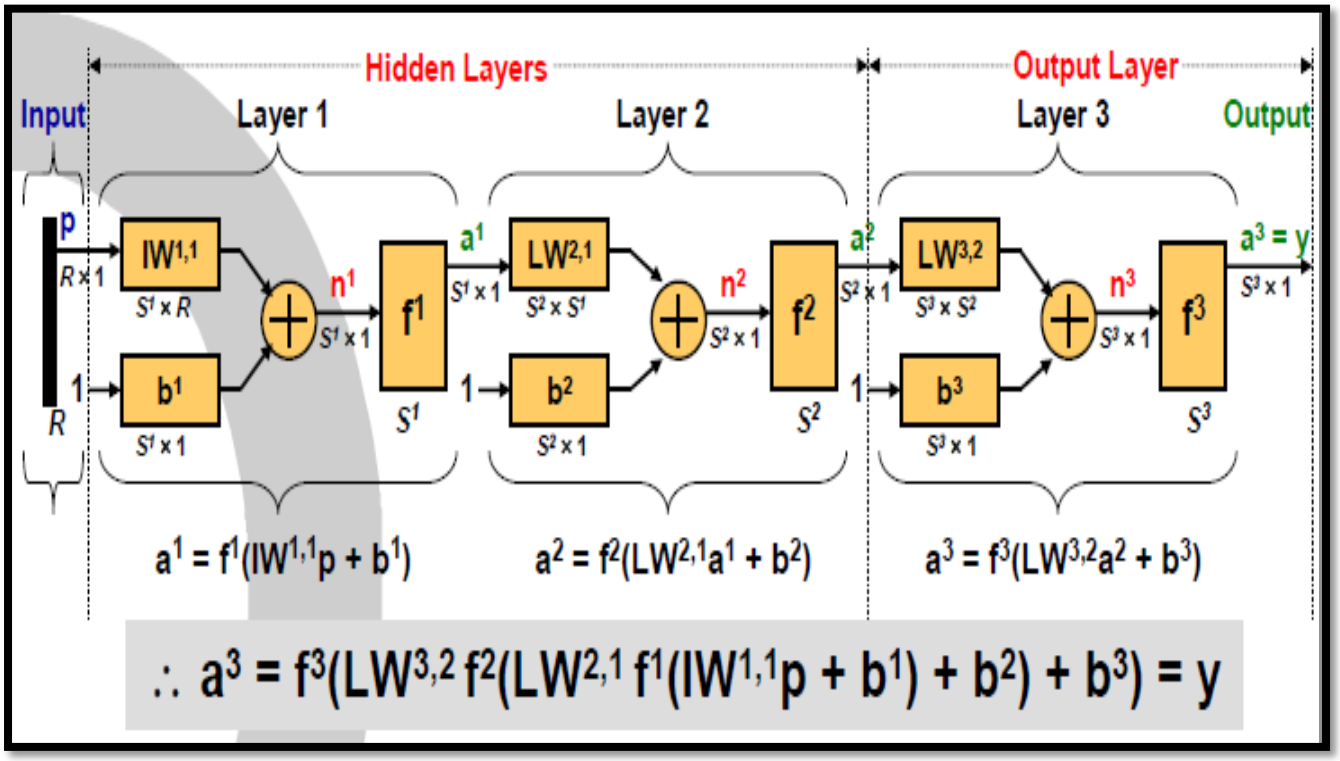
- We will use superscripts to identify the source (second index) and the destination (first index) for the various weights and other elements of the network.



- This network can be used as a general function approximator



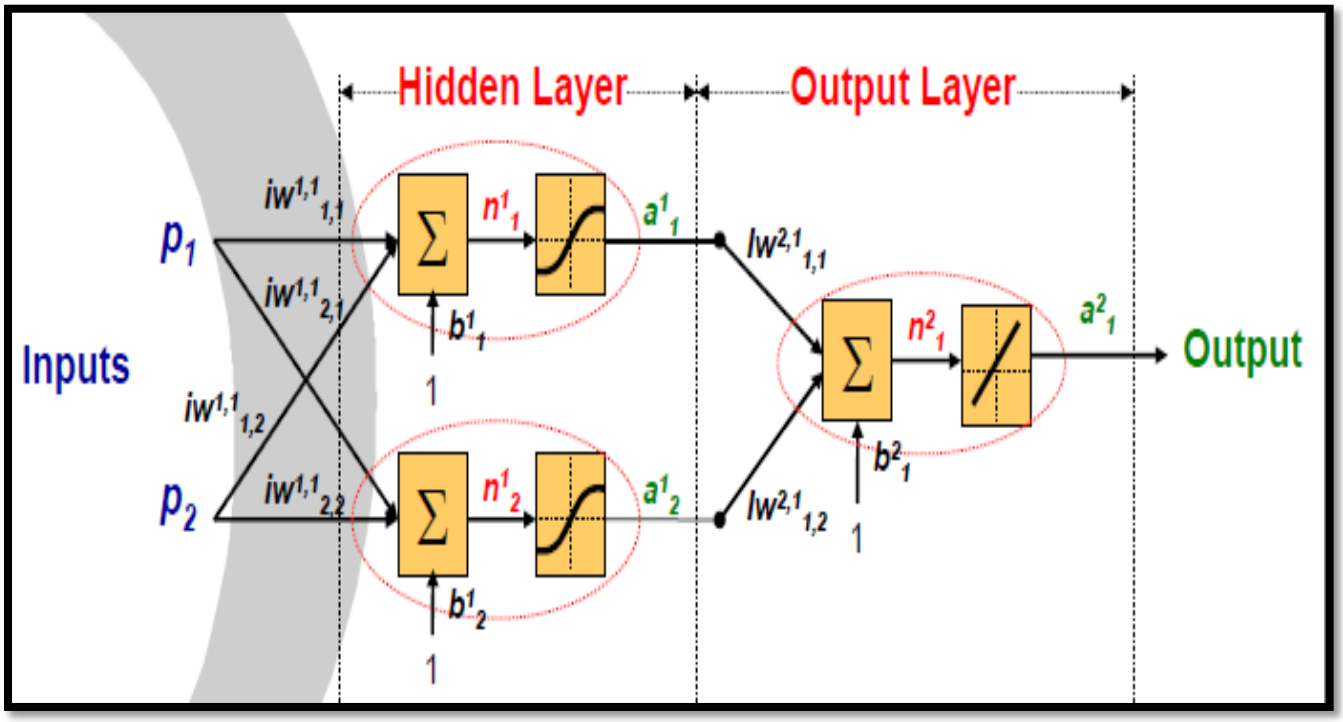
Example 1:
Multiple Layers of Neurons (Three-Layer Network, Abbreviated Notation)



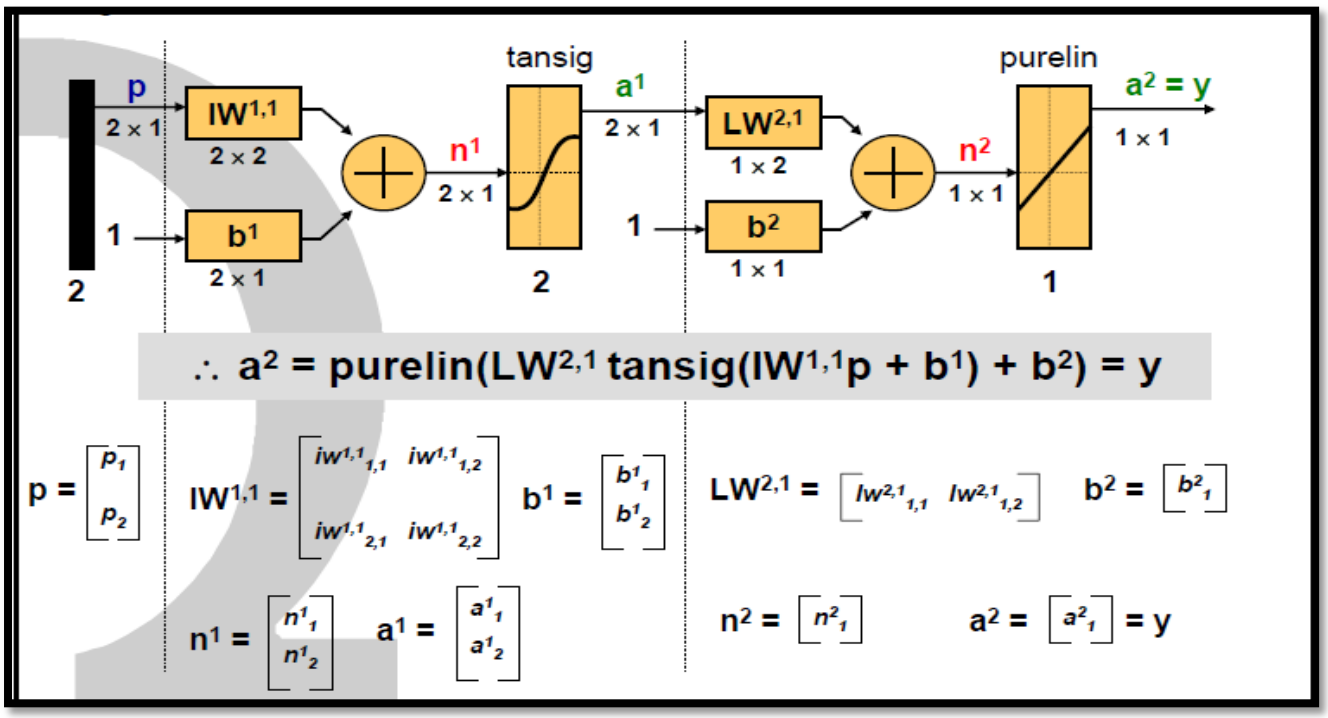
- S^1, S^2, S^3: Number of neurons in Layer 1, Layer 2, Layer 3 respectively
- $IW^{1,1}$: Input Weight matrix for connection from Input to Layer 1
- $LW^{2,1}$: Layer Weight matrix for connection from Layer 1 to Layer 2
- $LW^{3,2}$: Layer Weight matrix for connection from Layer 2 to Layer 3

Example 2:

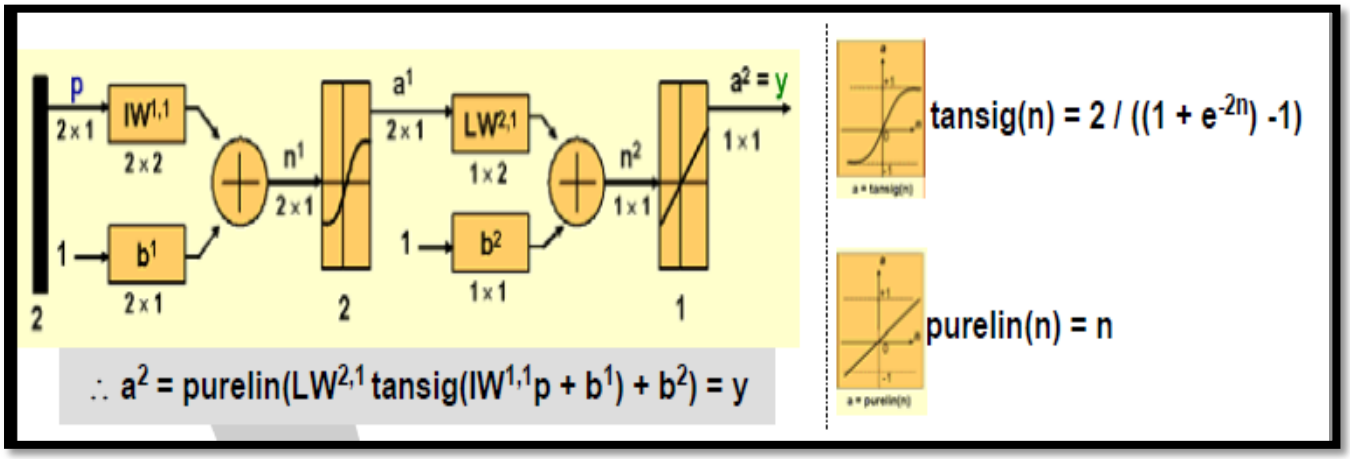
Neural network with 2 layers, 1st layer (hidden layer) consists of 2 neurons with *tangent-sigmoid* (**tansig**) transfer functions; 2nd layer (output layer) consists of 1 neuron with *linear* (**purelin**) transfer function.

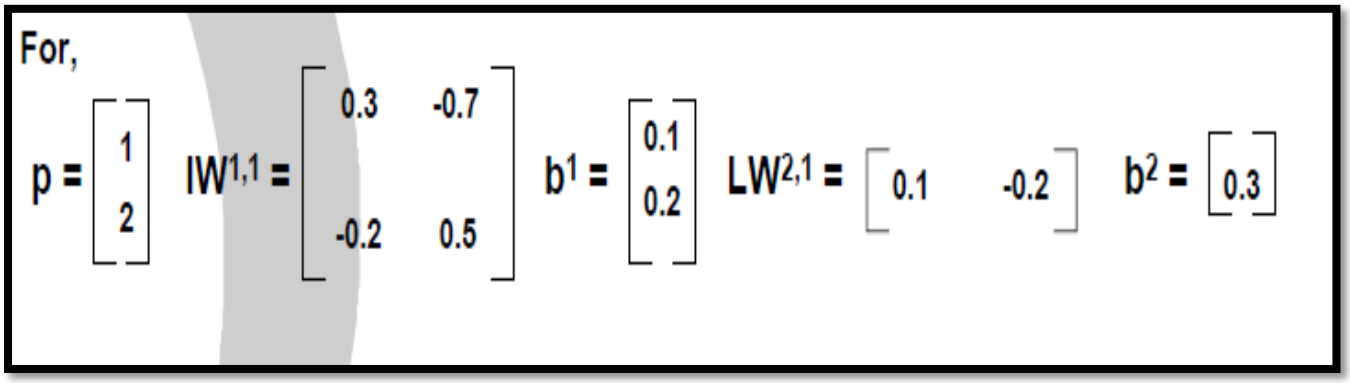


In MATLAB® abbreviated notation, the neural network is represented by the diagram below.



Example 3:





y = a2 = purelin(LW2,1 tansig(IW1,1p + b1) + b2)
= purelin([0.1 -0.2] × tansig([0.3 -0.7 ; -0.2 0.5] × [1 ; 2] + [0.1 ; 0.2]) + 0.3)
= purelin([0.1 -0.2] × tansig([-1 ; 1]) + 0.3)
= purelin(0.0715)
= 0.0715