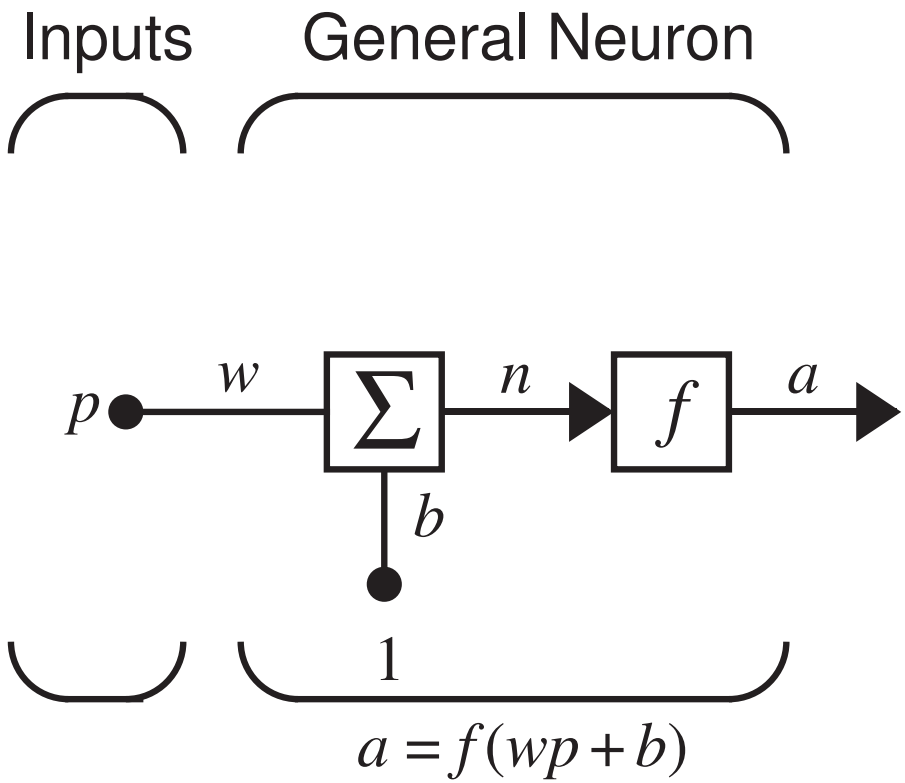
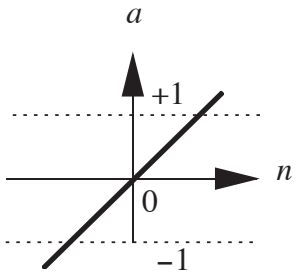
Multilayer Networks

Deep Learning

Basic network building block (neuron)

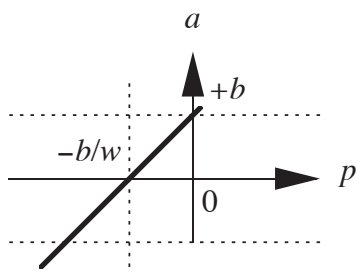


Transfer (activation) functions (1)



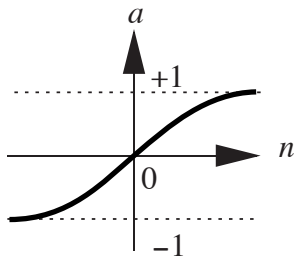
$a = purelin(n)$

Linear Function



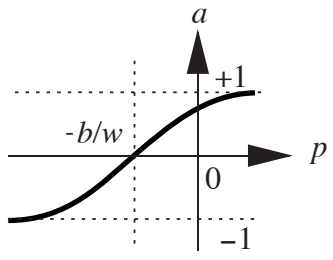
$a = purelin(wp + b)$

Single-Input *purelin* Neuron



$a = tansig(n)$

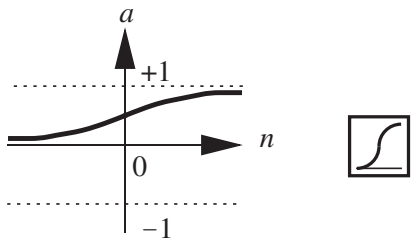
Tan-Sigmoid Function

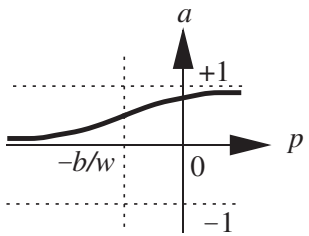


$a = tansig(wp + b)$

Single-Input *tansig* Neuron

Transfer (activation) functions (2)



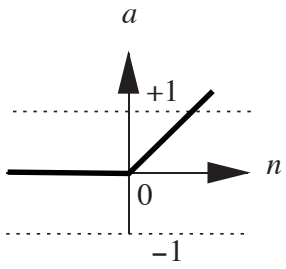


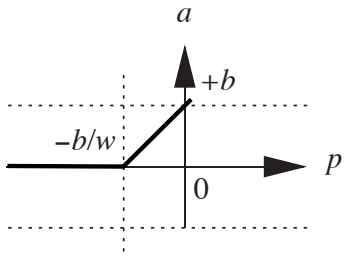
$a = logsig(n)$

Log-Sigmoid Transfer Function

$a = logsig(wp + b)$

Single-Input *logsig* Neuron





$a = poslin(n)$

Positive Linear Function

$a = poslin(wp + b)$

Single-Input *poslin* Neuron

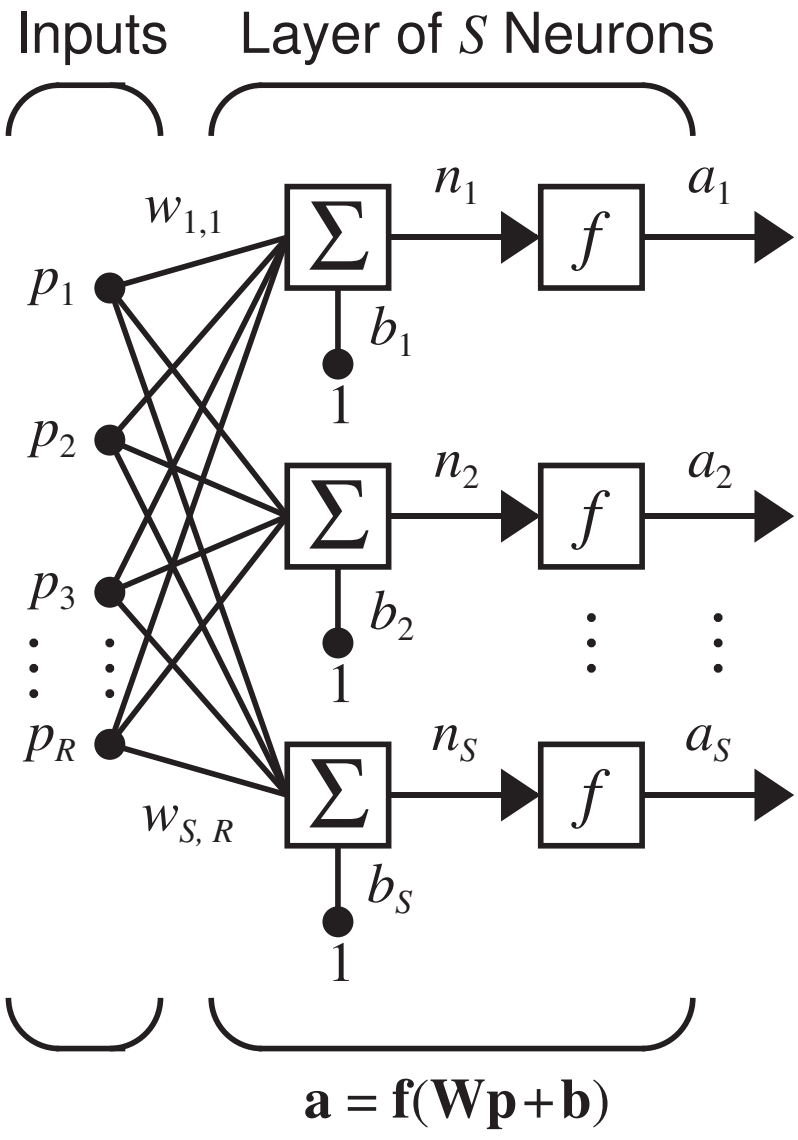
Transfer (activation) functions (3)

Softmax

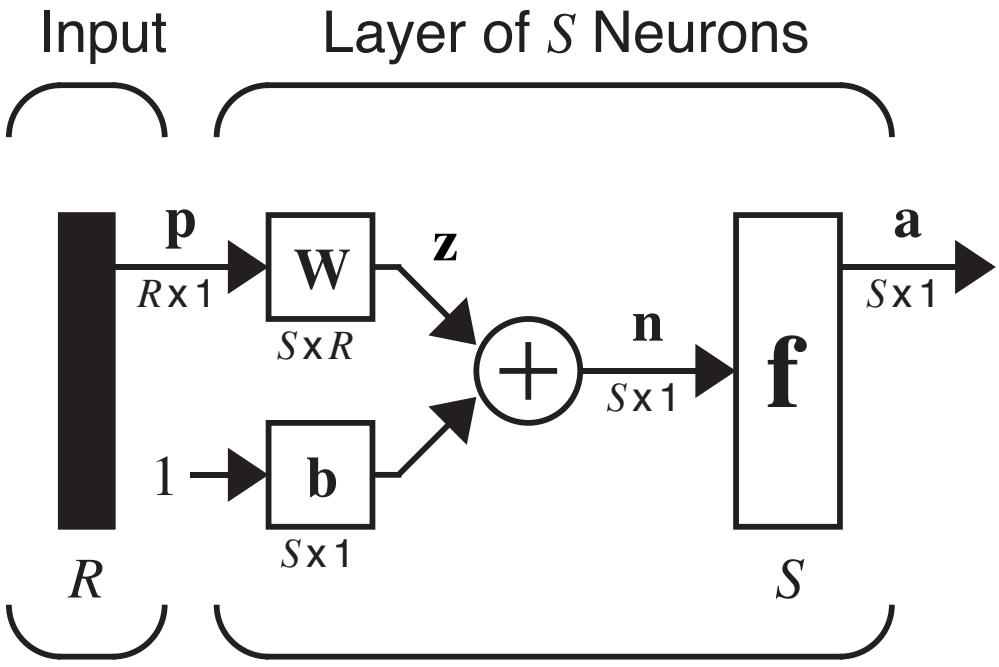
$$a_i = f_i(\mathbf{n}) = \frac{e^{n_i}}{\sum_{j=1}^{S} e^{n_j}}$$

Used at the output layer of a pattern recognition network with multiple output neurons.

Layer of neurons

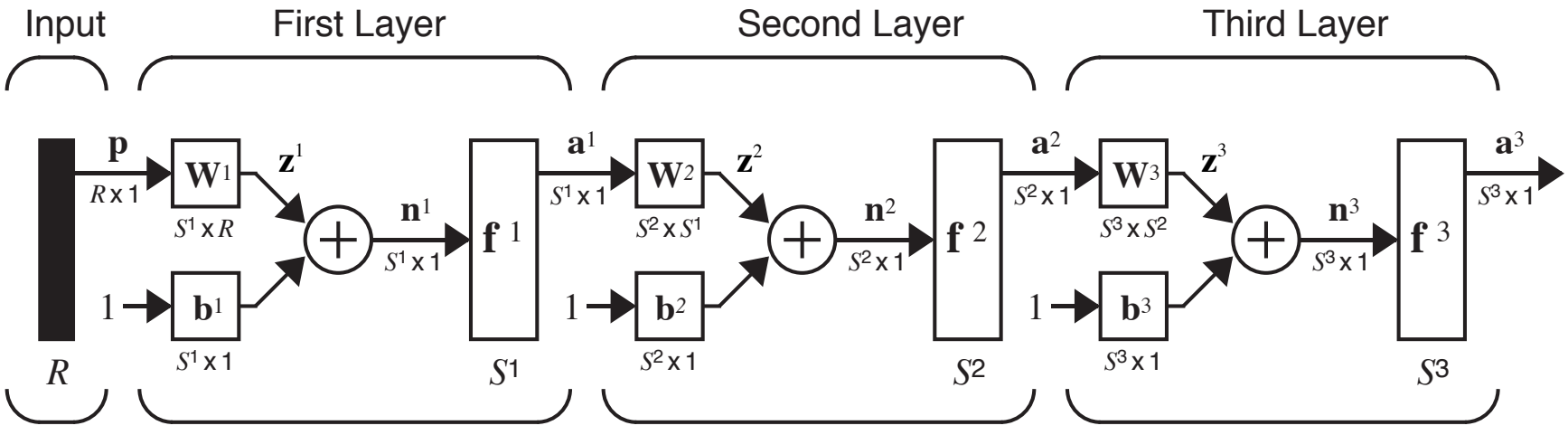


Matrix notation



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

Multiple layer network



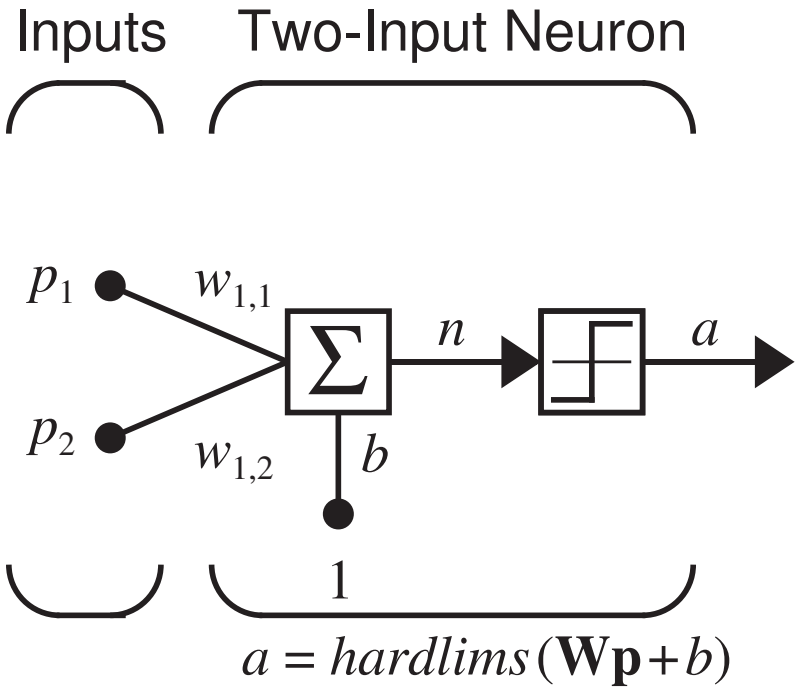
$$\mathbf{a}^1 = \mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1) \qquad \mathbf{a}^2 = \mathbf{f}^2(\mathbf{W}^2\mathbf{a}^1+\mathbf{b}^2) \qquad \mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{a}^2+\mathbf{b}^3)$$

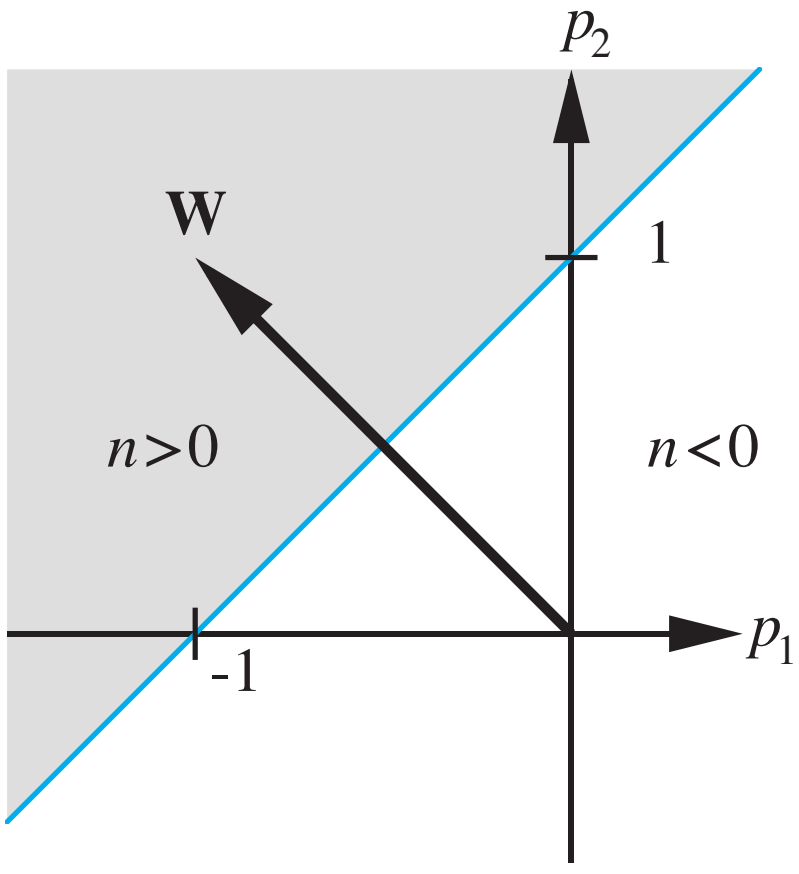
$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{f}^2(\mathbf{W}^2\mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1)+\mathbf{b}^2)+\mathbf{b}^3)$$

Single layer network decision boundary

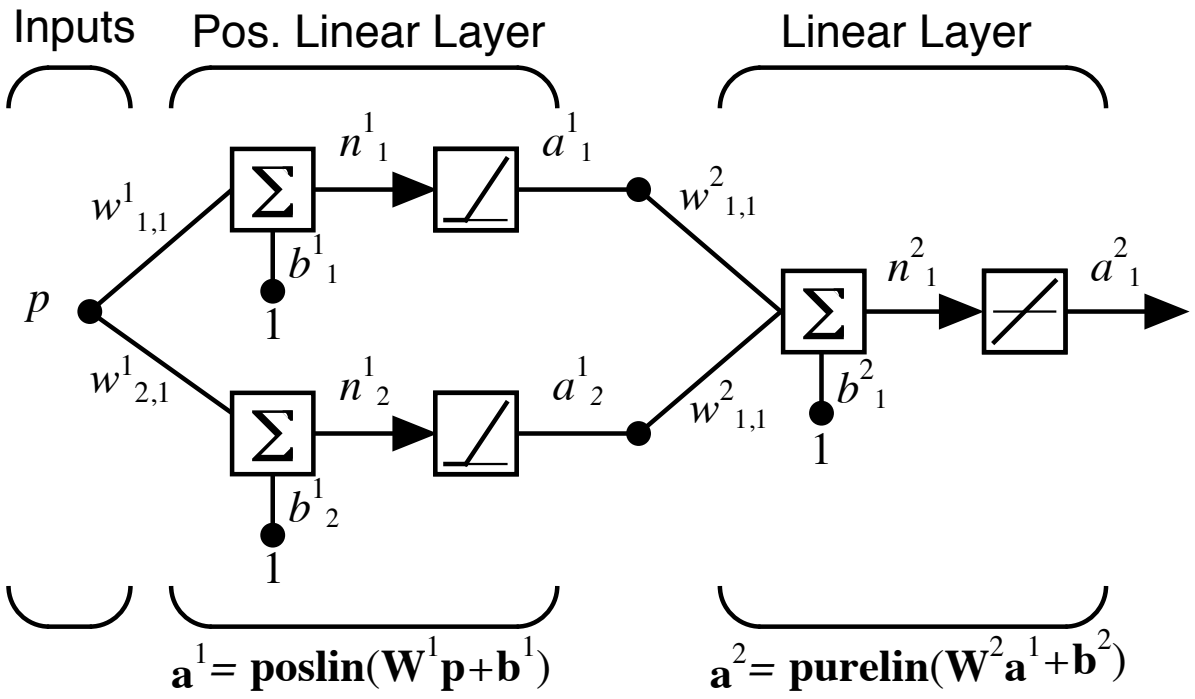
Decision boundary

$$n = \mathbf{W}\mathbf{p} + b == 0$$





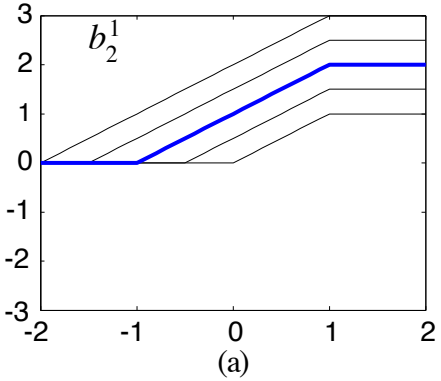
Poslin network

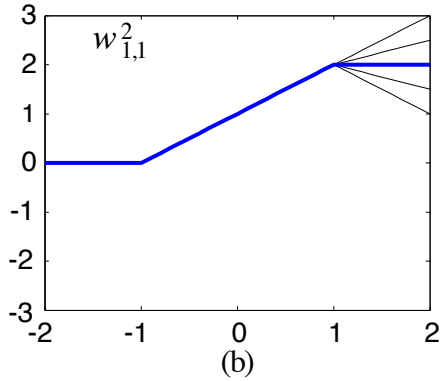


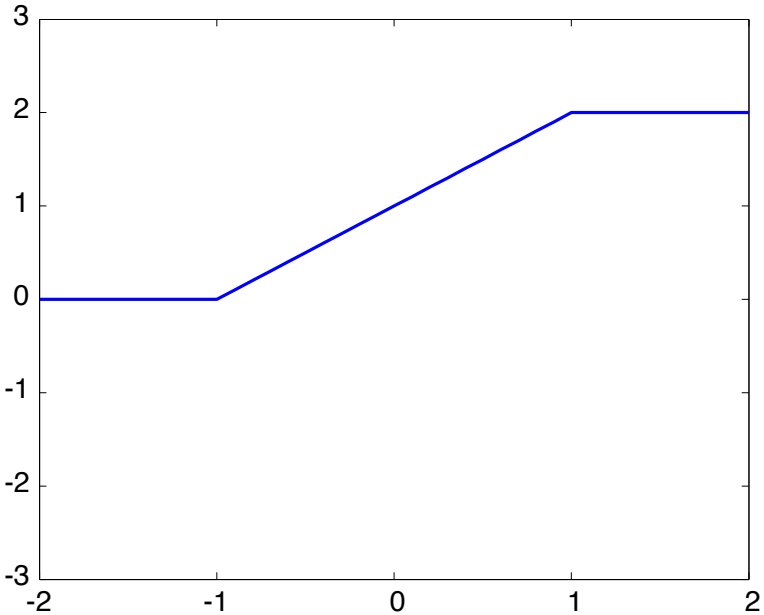
Poslin network function

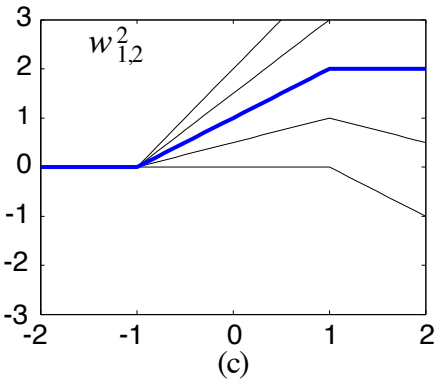
$$\mathbf{W}^1 = \begin{bmatrix} 1 & 1 \end{bmatrix}^T, \mathbf{b}^1 = \begin{bmatrix} -1 & 1 \end{bmatrix}^T$$

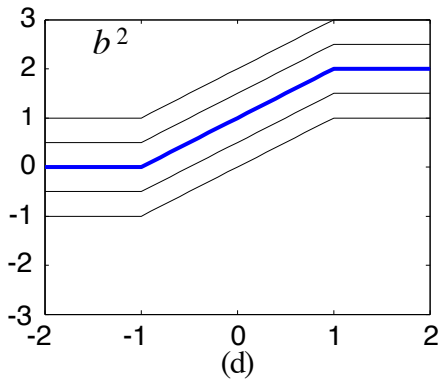
$$\mathbf{W}^2 = \begin{bmatrix} -1 & 1 \end{bmatrix}, \mathbf{b}^2 = [0]$$



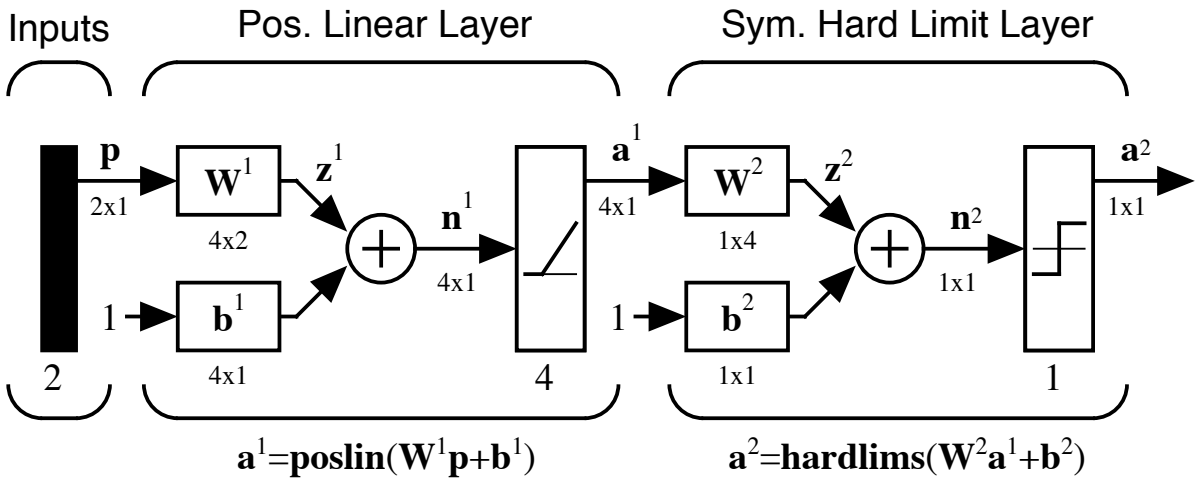








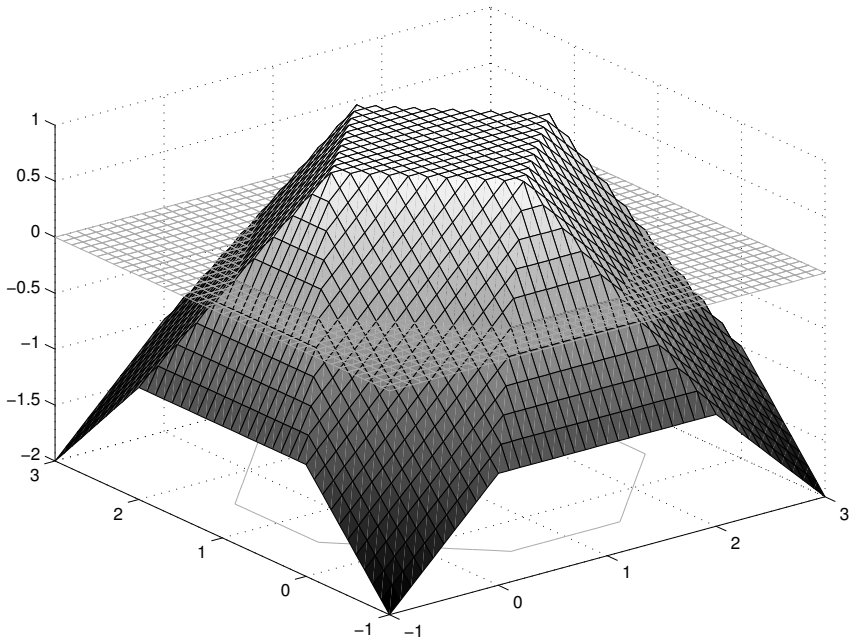
2D poslin network

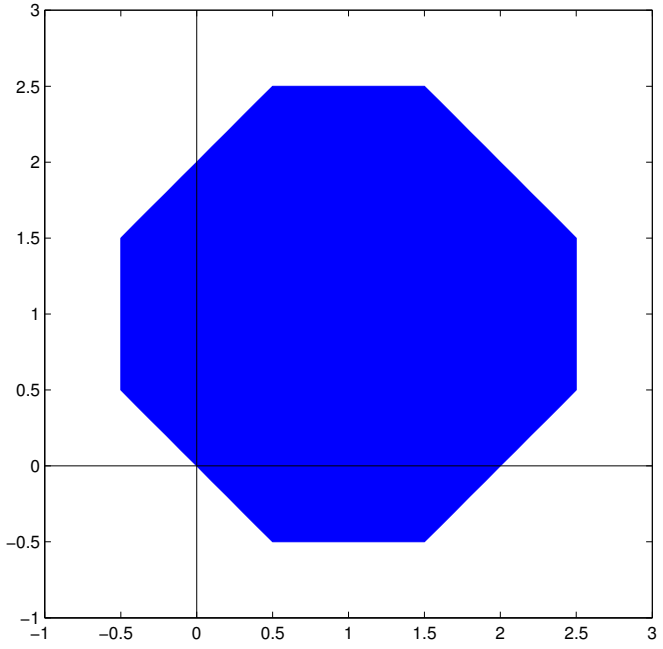


$$\mathbf{W}^1 = \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}^T, \mathbf{b}^1 = \begin{bmatrix} -1 & 3 & 1 & 1 \end{bmatrix}^T$$

$$\mathbf{W}^2 = \begin{bmatrix} -1 & -1 & -1 & -1 \end{bmatrix}, \mathbf{b}^2 = [5]$$

2D Poslin network surface and decision boundary





Why do we need deep networks?

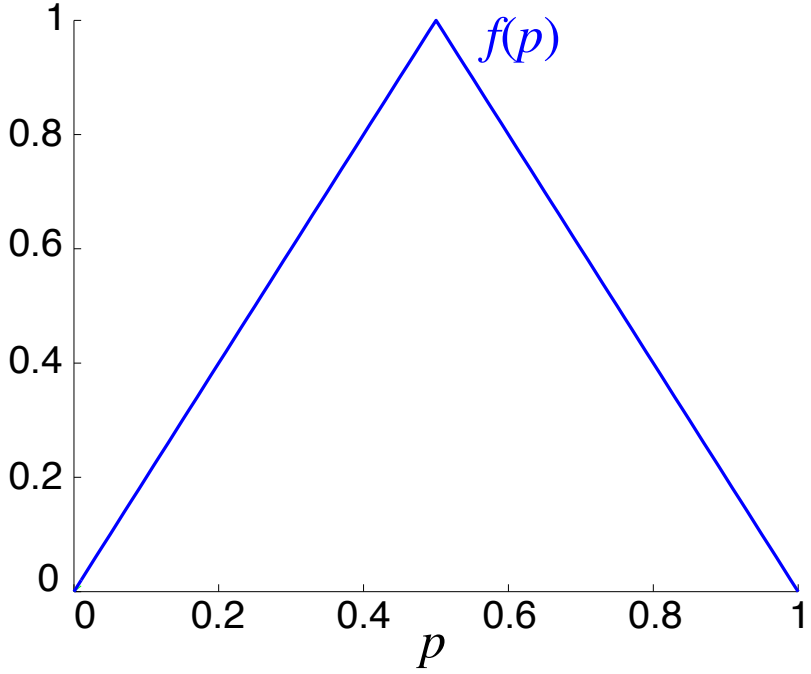
- Theoretically, two-layer networks (with sufficient hidden neurons) can approximate any practical function or decision region.
- In practice, deep networks have been able to solve problems that two-layer networks were not previously able to.
- The theoretical explanation of this is not complete.
- We will describe a couple of initial explanations.
- The idea is that deeper networks can produce more efficient realizations of some complex functions than two layer networks can.

Cascading layers (1)

- Consider again the earlier two-layer network, but change the weights.

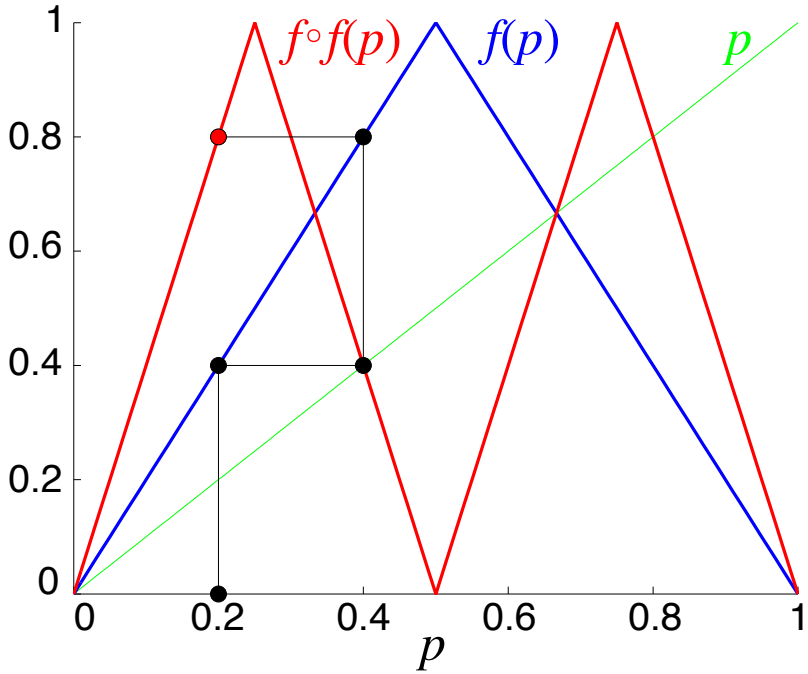
$$\mathbf{W}^1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mathbf{b}^1 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}, \mathbf{W}^2 = \begin{bmatrix} 2 & -4 \end{bmatrix}, \mathbf{b}^2 = [0]$$

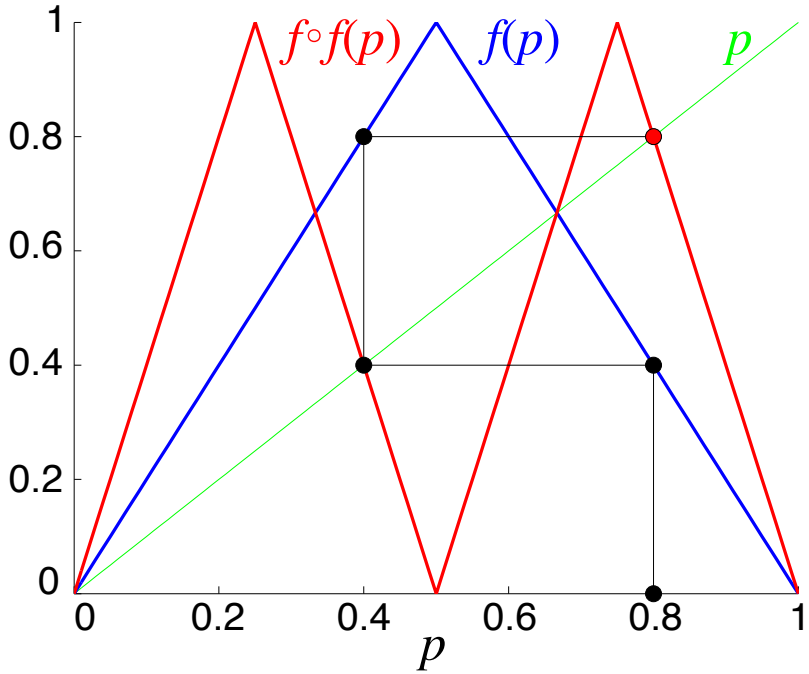
- We get the following function.



Cascading layers (2)

- Now cascade the function with itself: $f \circ f(p) = f(f(p))$.
- We get the following function (in red).
- By adding a single layer we have doubled the complexity.



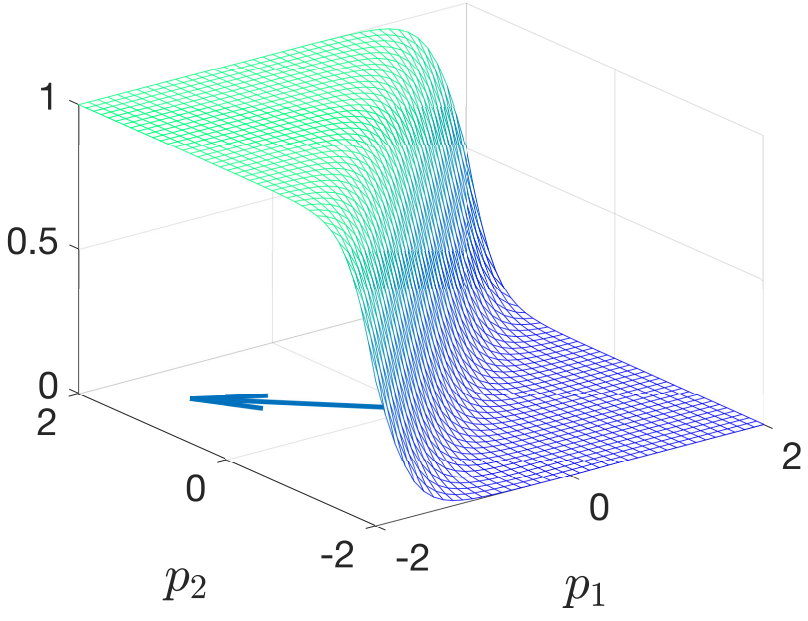


Cascading layers (3)

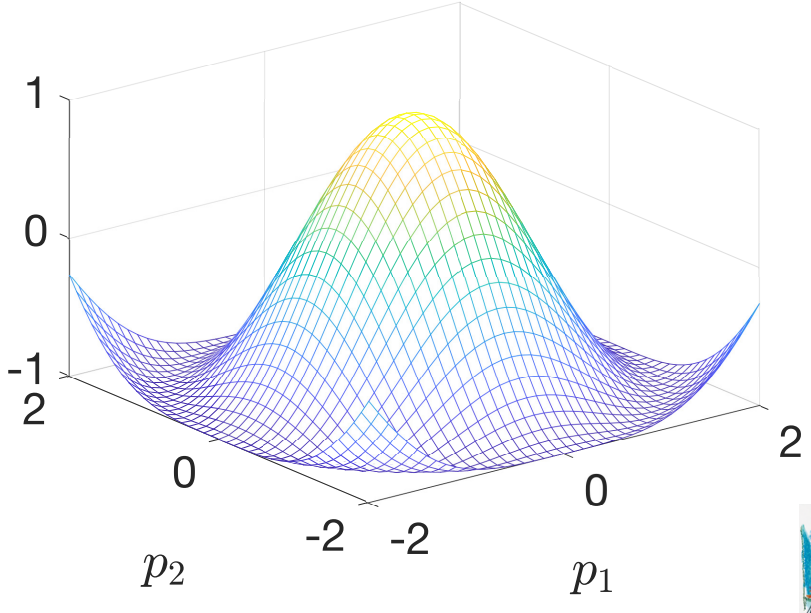
- Each time we add a layer (a linear increase in the number of neurons), we double the number of neurons that would be required for a two layer network to implement the same function.
- Two-layer networks can create any function that can be created with a deep network.
- The number of neurons required in the two-layer network increase geometrically.
- The number of neurons increase linearly in the equivalent deep network.

Implementing radial functions

- A sigmoid neuron response does not change orthogonal to the weight direction, which creates a ridge function.
- It is difficult to approximate a radial function with a sum of ridges (two-layer network).
- With the addition of a third layer, a 2-4-2-1 network can easily make an accurate approximation.



Sigmoid Neuron Response



Radial Function