Neural Networks
[RN2] Sec 20.5
[RN3] Sec 20.5

CS 486/686

University of Waterloo

Lecture 19: July 7, 2015

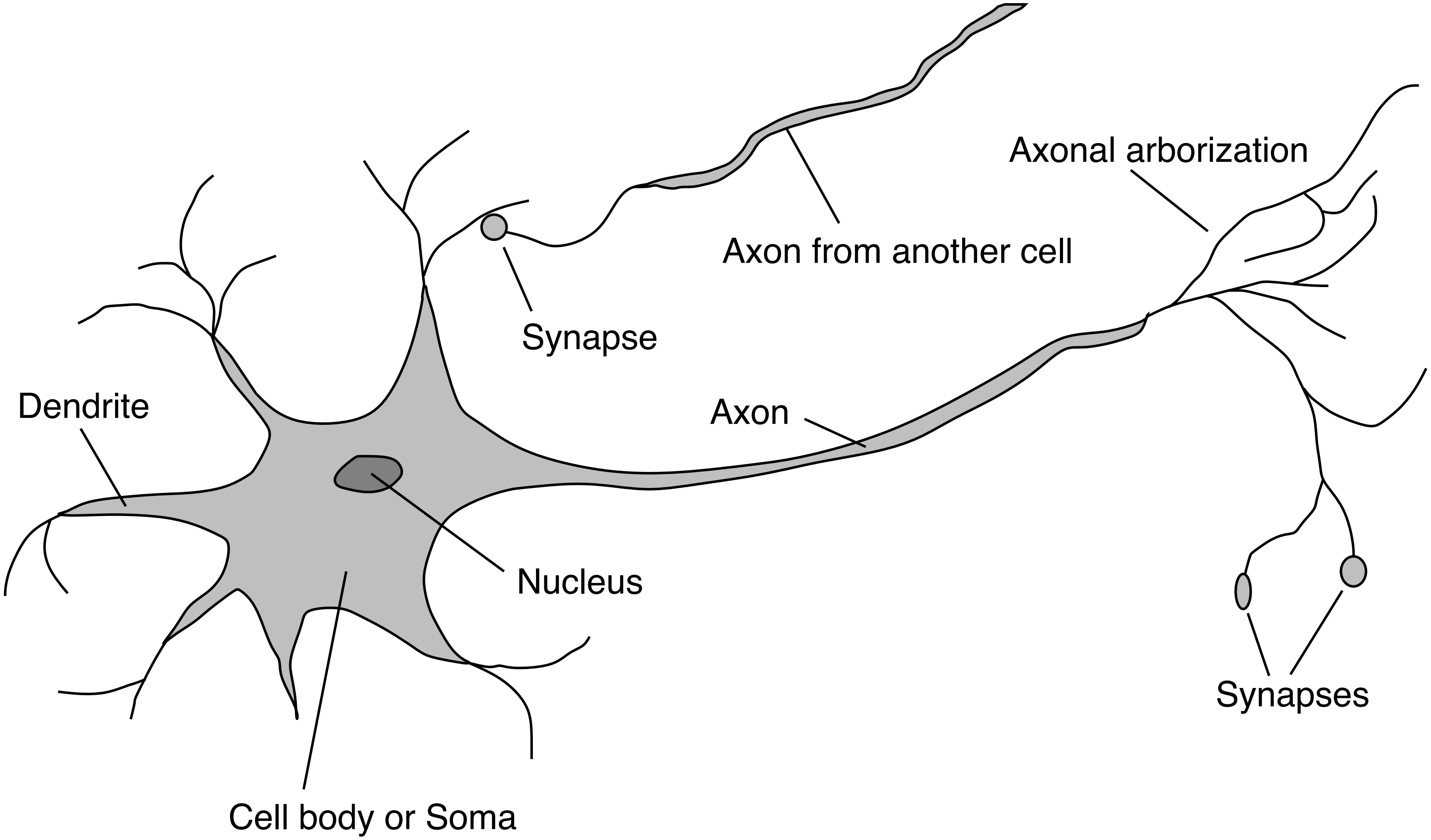
Outline

- Neural networks
  - Perceptron
  - Supervised learning algorithms for neural networks

Brain

- Seat of human intelligence
- Where memory/knowledge resides
- Responsible for thoughts and decisions
- Can learn
- Consists of nerve cells called neurons

Neuron


Axonal arborization
Axon from another cell
Synapse
Dendrite
Axon
Nucleus
Synapses
Cell body or Soma

Comparison

- Brain
  - Network of neurons
  - Nerve signals propagate in a neural network
  - Parallel computation
  - Robust (neurons die everyday without any impact)
- Computer
  - Bunch of gates
  - Electrical signals directed by gates
  - Sequential and parallel computation
  - Fragile (if a gate stops working, computer crashes)

Artificial Neural Networks

- Idea: mimic the brain to do computation

- Artificial neural network:
  - Nodes (a.k.a. units) correspond to neurons
  - Links correspond to synapses

- Computation:
  - Numerical signal transmitted between nodes corresponds to chemical signals between neurons
  - Nodes modifying numerical signal correspond to neurons firing rate

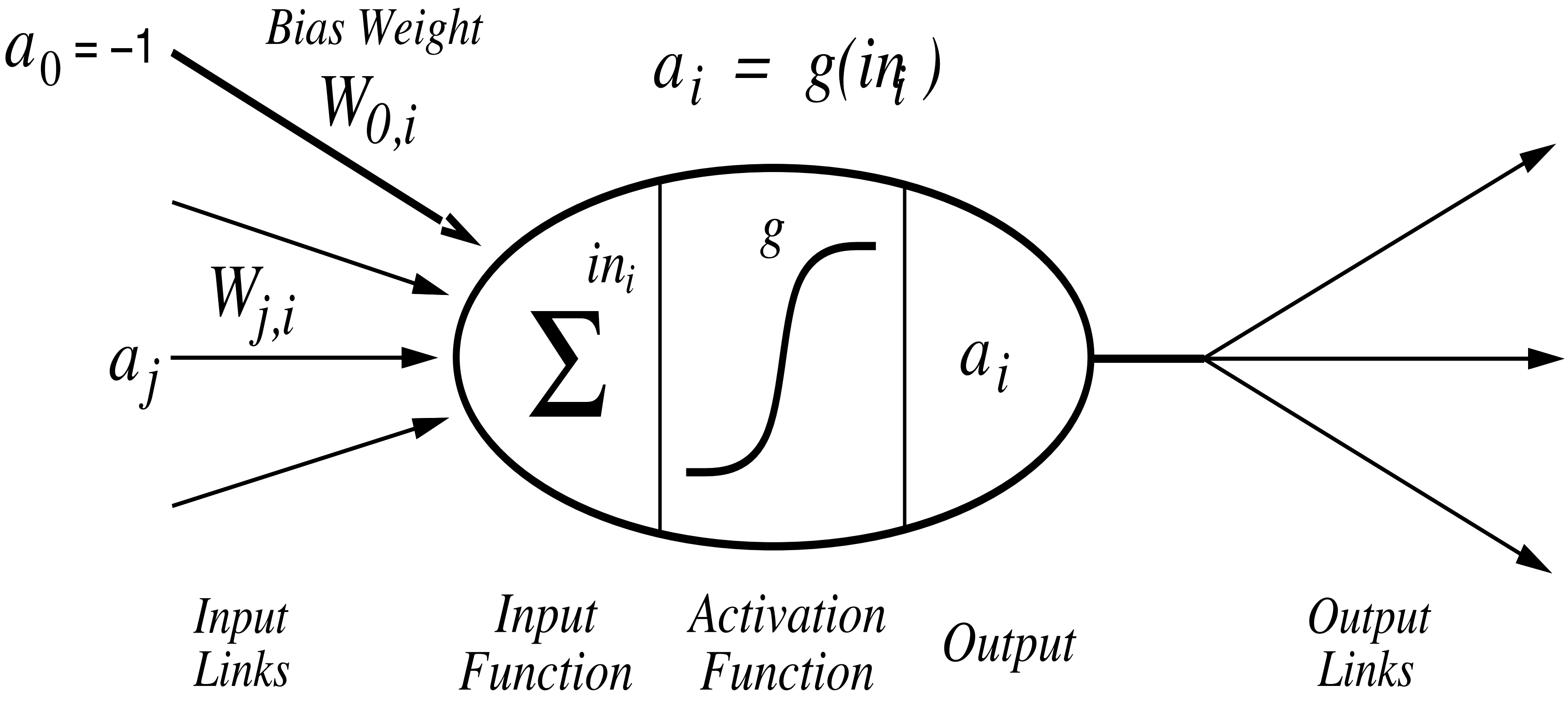
ANN Unit

- For each unit i:

- Weights: W_{ji}
  - Strength of the link from unit j to unit i
  - Input signals a_j weighted by W_{ji} and linearly combined: $in_i = \Sigma_j W_{ji} a_j$

- Activation function: g
  - Numerical signal produced: $a_i = g(in_i)$

ANN Unit


$a_0 = -1$
Bias Weight
$W_{0,i}$
$a_i = g(in_i)$
$W_{j,i}$
a_j
in_i
Σ
g
a_i
Input Links
Input Function
Activation Function
Output
Output Links

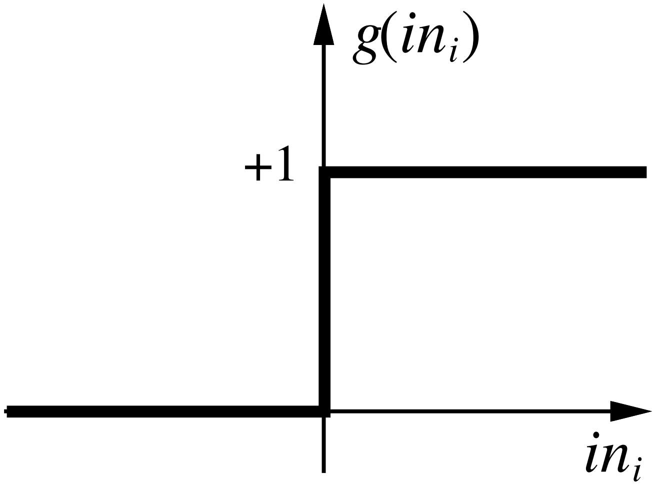
Activation Function

- Should be nonlinear
  - Otherwise network is just a linear function

- Often chosen to mimic firing in neurons
  - Unit should be "active" (output near 1) when fed with the "right" inputs
  - Unit should be "inactive" (output near 0) when fed with the "wrong" inputs

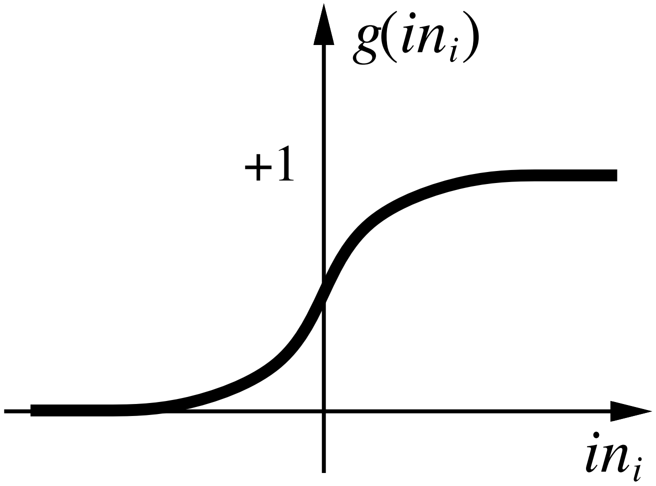
Common Activation Functions

Threshold



(a)

Sigmoid

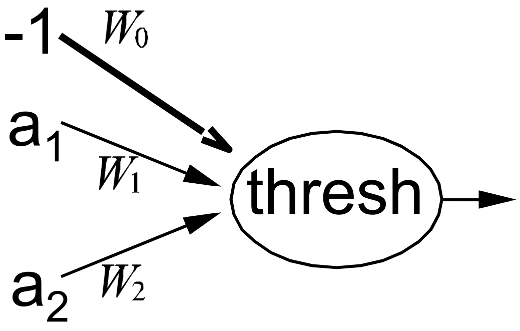


(b)

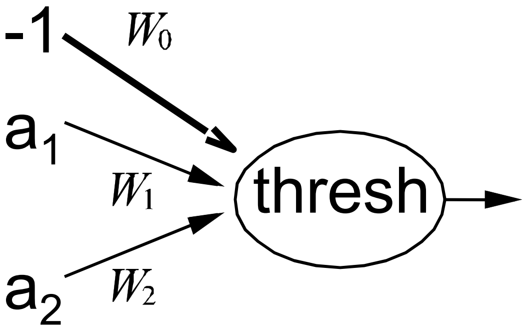
$$g(x) = 1/(1+e^{-x})$$

Logic Gates

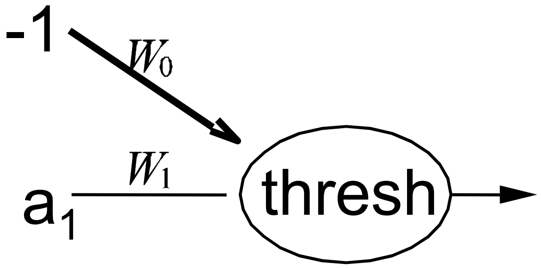
- McCulloch and Pitts (1943)
  - Design ANNs to represent Boolean fns
- What should be the weights of the following units to code AND, OR, NOT ?



AND



OR



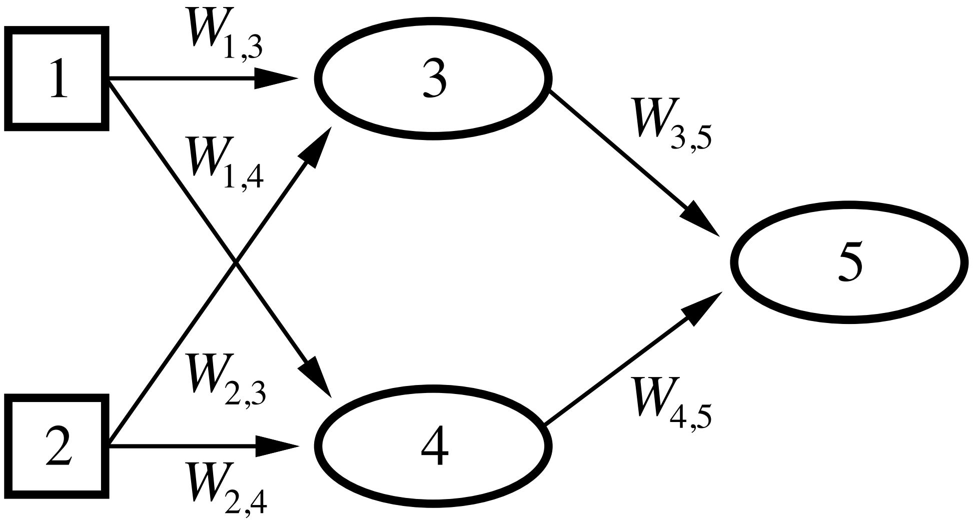
NOT

Network Structures

- Feed-forward network
  - Directed acyclic graph
  - No internal state
  - Simply computes outputs from inputs
- Recurrent network
  - Directed cyclic graph
  - Dynamical system with internal states
  - Can memorize information

Feed-forward network

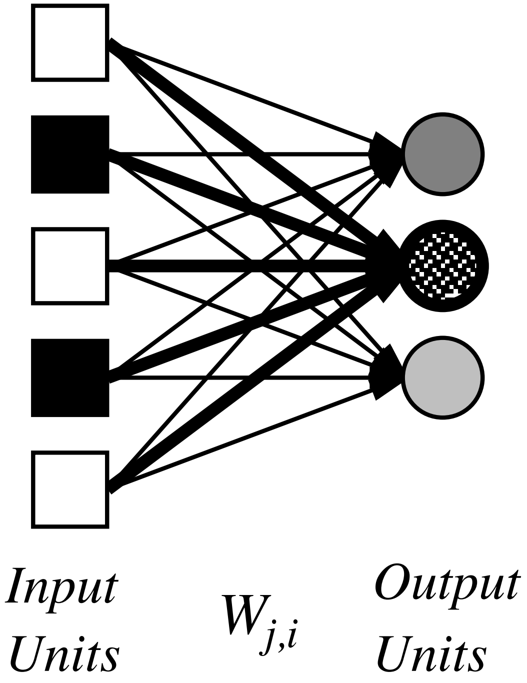
- Simple network with two inputs, one hidden layer of two units, one output unit



$$
\begin{aligned}
a_5 &= g(W_{3,5}a_3 + W_{4,5}a_4) \\
&= g(W_{3,5}g(W_{1,3}a_1 + W_{2,3}a_2) + W_{4,5}g(W_{1,4}a_1 + W_{2,4}a_2))
\end{aligned}
$$

Perceptron

- Single layer feed-forward network



Supervised Learning

- Given list of <input,output> pairs
- Train feed-forward ANN
  - To compute proper outputs when fed with inputs
  - Consists of adjusting weights W_{ji}
- Simple learning algorithm for threshold perceptrons

Threshold Perceptron Learning

- Learning is done separately for each unit
  - Since units do not share weights

- Perceptron learning for unit i:
  - For each <inputs,output> pair do:
    - Case 1: correct output produced

      $\forall_j \; W_{ji} \leftarrow W_{ji}$
    - Case 2: output produced is 0 instead of 1

      $\forall_j \; W_{ji} \leftarrow W_{ji} + a_j$
    - Case 3: output produced is 1 instead of 0

      $\forall_j \; W_{ji} \leftarrow W_{ji} - a_j$
  - Until correct output for all training instances

Threshold Perceptron Learning

- Dot products: $a \bullet a \geq 0$ and $-a \bullet a \leq 0$
- Perceptron computes

  1 when $a \bullet W = \Sigma_j \, a_j W_{ji} \geq 0$

  0 when $a \bullet W = \Sigma_j \, a_j W_{ji} < 0$
- If output should be 1 instead of 0 then

  $W \leftarrow W + a$ since $a \bullet (W+a) \geq a \bullet W$
- If output should be 0 instead of 1 then

  $W \leftarrow W - a$ since $a \bullet (W-a) \leq a \bullet W$

Threshold Perceptron Hypothesis Space

- Hypothesis space h_W:
  - All binary classifications with parameters W s.t.

    $a \bullet W \geq 0 \rightarrow 1$

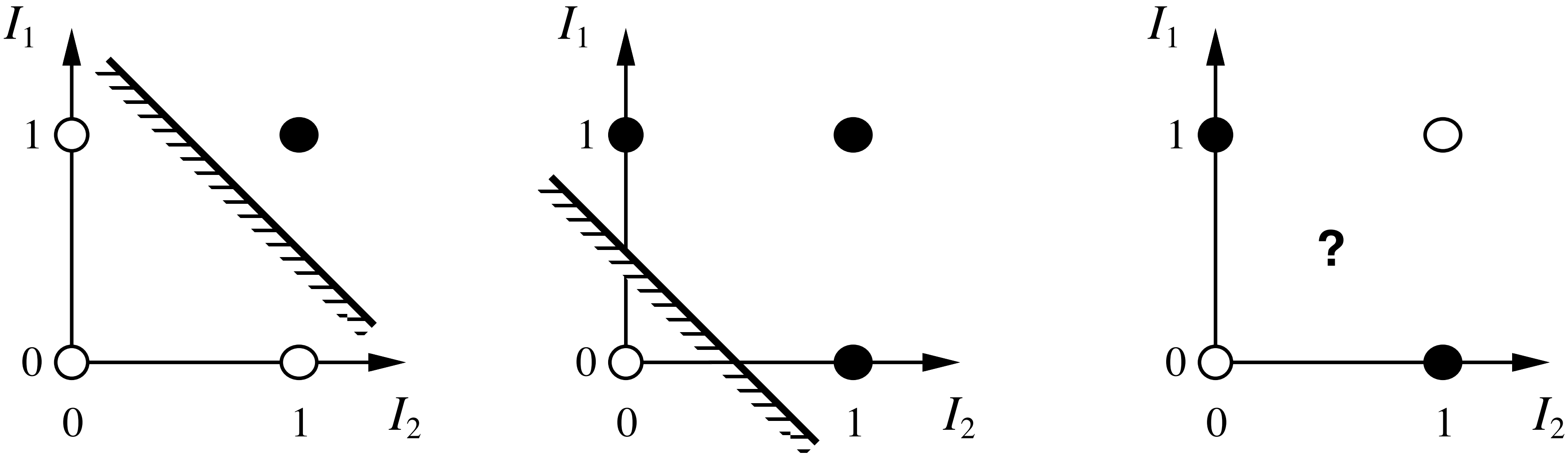
    $a \bullet W < 0 \rightarrow 0$

- Since $a \bullet W$ is linear in W, perceptron is called a linear separator

- Theorem: threshold perceptron learning converges iff the data is linearly separable.

Threshold Perceptron Hypothesis Space

- Are all Boolean gates linearly separable?



(a) I_1 **and** I_2

(b) I_1 **or** I_2

(c) I_1 **xor** I_2

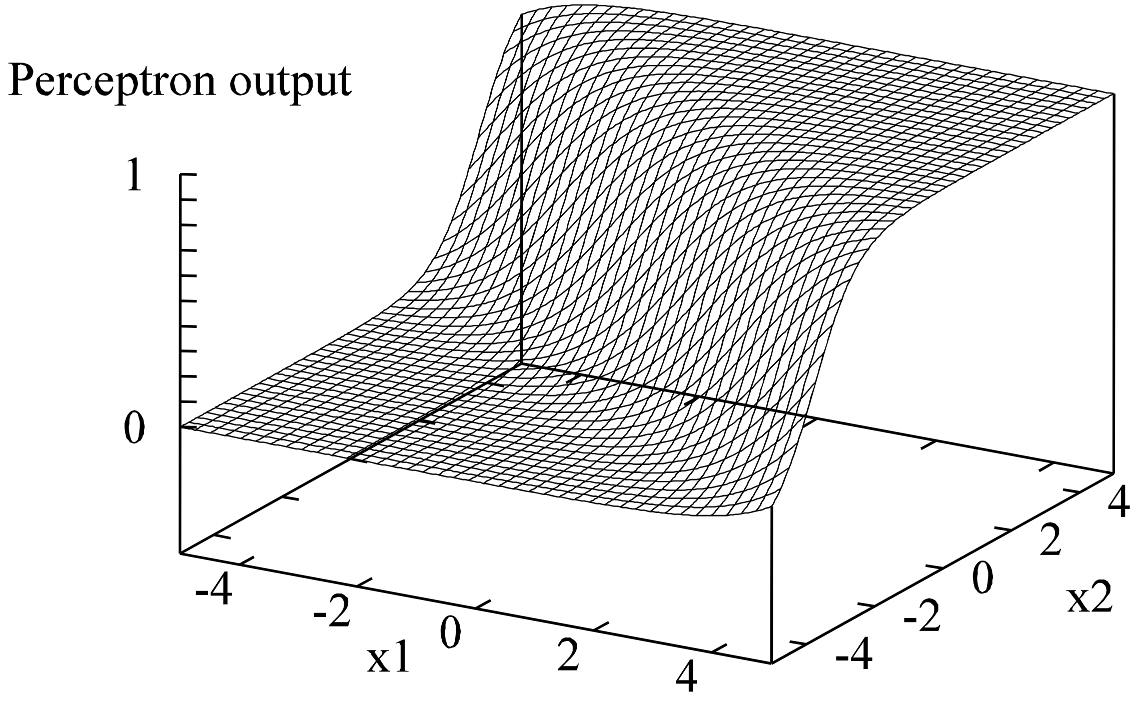
Example: Threshold Perceptron Learning

- AND gate   Data: {(0,0)→0, (0,1)→0, (1,0)→0, (1,1)→1}

Inputs x_0, x_1, x_2	**Output** y	**Weights** W_0, W_1, W_2	**Prediction** $h_W(x)$	**error**
1,0,0	0	0.1, -0.2, 0.3	1	yes
1,0,1	0			
1,1,0	0			
1,1,1	1			
1,0,0	0			
1,0,1	0			
1,1,0	0			
1,1,1	1			
1,0,0	0			
1,0,1	0			
1,1,0	0			
1,1,1	1			
1,0,0	0			
1,0,1	0			

Sigmoid Perceptron

- Represent "soft" linear separators



Sigmoid Perceptron Learning

- Formulate learning as an optimization search in weight space
  - Since g differentiable, use gradient descent

- Minimize squared error:

  $E = 0.5\ Err^2 = 0.5\ (y - h_W(\mathbf{x}))^2$

    - $\mathbf{x}$: input
    - y: target output
    - $h_W(\mathbf{x})$: computed output

Perceptron Error Gradient

- $E = 0.5\ Err^2 = 0.5\ (y - h_{\mathbf{W}}(\mathbf{x}))^2$

- $\partial E/\partial W_j = Err\ \partial Err/\partial W_j$
  $= Err\ \partial(y - g(\Sigma_j\ W_j x_j))/\partial W_j$
  $= -Err\ g'(\Sigma_j\ W_j x_j)\ x_j$

- When g is sigmoid fn, then $g' = g(1-g)$

Perceptron Learning Algorithm

- Perceptron-Learning(examples,network)
  - Repeat
    - For each e in examples do

```
in ← $\Sigma_j W_j x_j[e]$
Err ← $y[e] - g(in)$
$W_j$ ← $W_j + \alpha$ Err $g'(in)\, x_j[e]$
```

  - Until some stopping criteria satisfied
  - Return learnt network

- N.B. α is a learning rate corresponding to the step size in gradient descent

Multilayer Feed-forward Neural Networks

- Perceptron can only represent (soft) linear separators
  - Because single layer

- Need multiple layers to represent more complicated separators