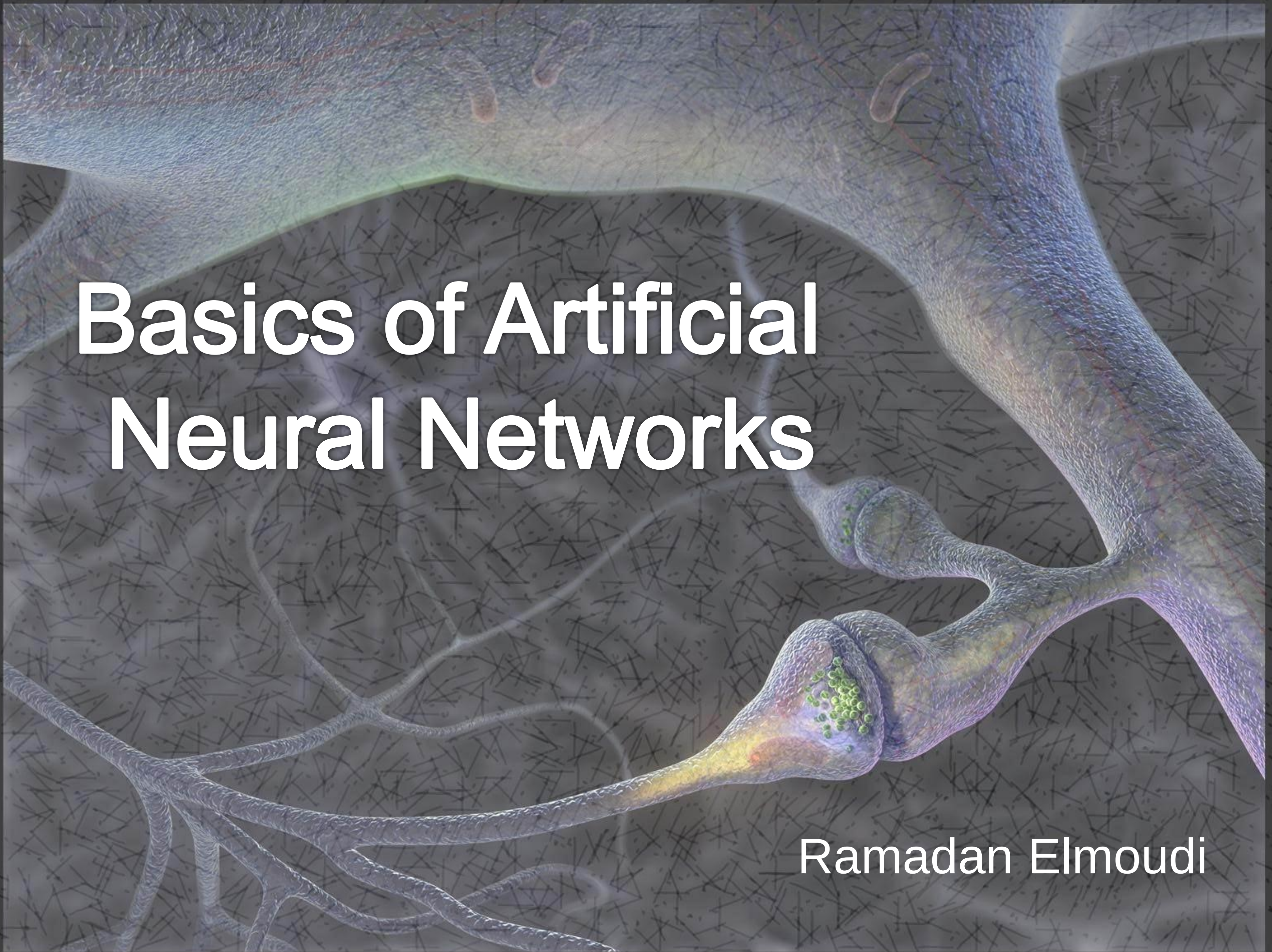
# IEEE Practitioner Tutorial

# Artificial Neural Networks for the Smart Grid Control

Ramadan Elmoudi
Ph.D Candidate
Electrical Engineering Department
University at Buffalo

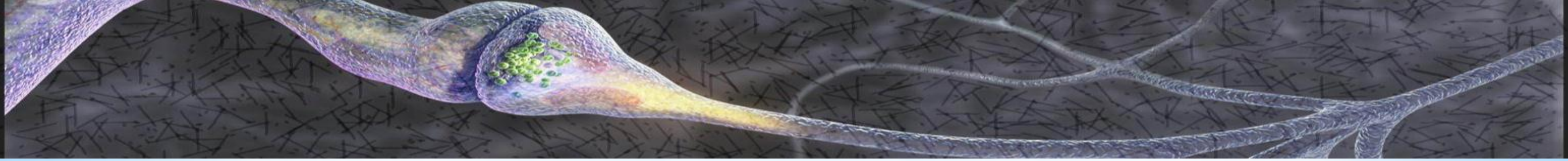
# Tutorial Outline

- Basics of Artificial Neural Networks
- Simulation of Artificial Neural Networks
- Case studies about the Applications of ANN in Smart Grids



# Basics of Artificial Neural Networks

Ramadan Elmoudi

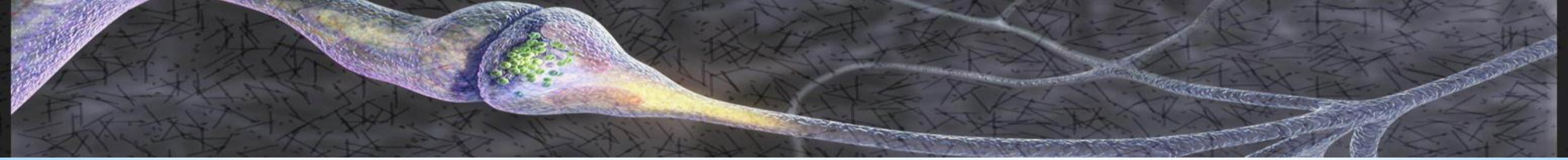


# One Minute QUIZ!

TH15 M3554G3 53RV35 TO PR0V3 H0W 0UR M1ND5 C4N D0 4M4Z1NG TH1NG5! 1MPR3551V3 TH1NG5! 1N TH3 B3G1NN1NG 1T WA5 H4RD BUT NOW, ON TH15 LIN3 YOUR M1ND 1S R34D1NG 1T 4UT0M4T1C4LLY W1TH OUT 3V3N TH1NK1NG 4B0UT 1T, B3 PROUD! 0NLY C34RT41N P30PL3 C4N R3AD TH15.

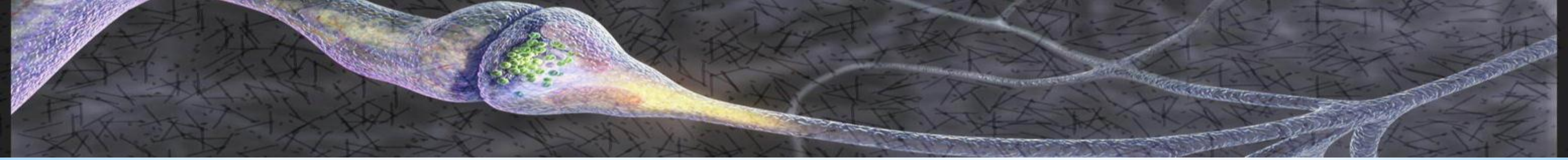
# Outline

- Definitions and Background
- Biological and Artificial Neuron
- Activation Functions and ANN layers
- Types of ANN
- Training of ANN
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- Applications of ANN



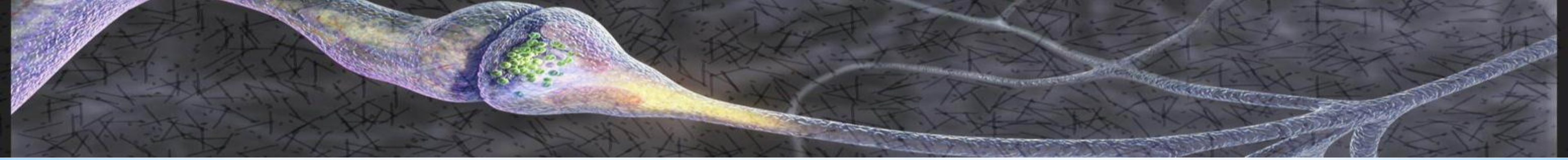
# Outline

- Definitions and Background
- Biological and Artificial Neuron
- Activation Functions and ANN layers
- Types of ANN
- Training of ANN
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- Applications of ANN



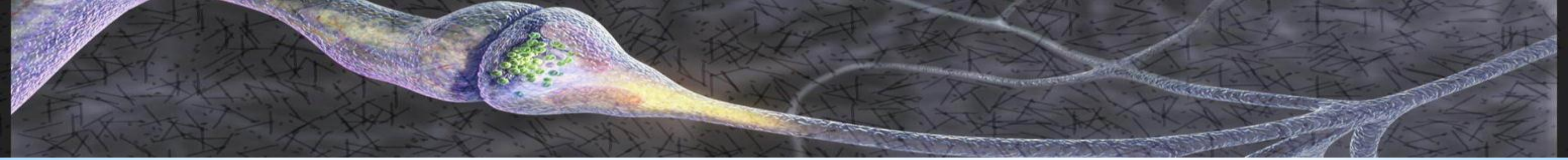
# Definitions

- **Neuron** is an excitable cell that process and transmits information by electrochemical signaling.
- **Neural Networks** are networks of neurons “brain”.
- **Artificial Neuron** is a crude approximation of the neuron that found in brain.
- **Artificial Neural Networks** are parallel computational system consisting of many simple processing elements connected together to solve a specific task.



# Why ANN worth Studying

- They are extremely powerful computational devices.
- Massive parallelism makes them very efficient.
- They can learn and generalize from training data.
- They are particularly fault tolerant.
- They are very noise tolerant.
- The ANN is itself nonlinear so it can model almost all nonlinear systems.
- Mathematical model of the system is not needed; since ANN generate the model from the input and output data.



# A brief History

Long history of development.

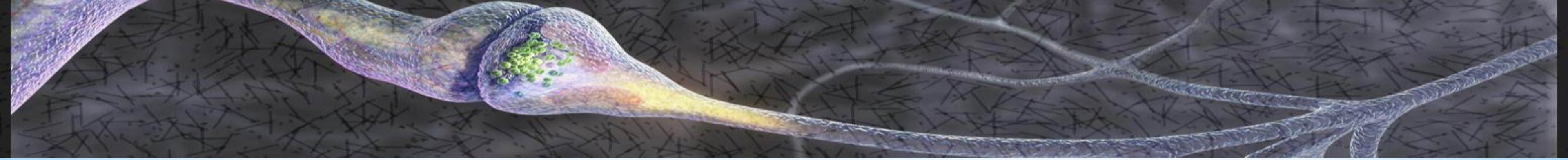
**1943** McCullach & Pitts[1] outlined the first formal model of an elementary computing neuron

**1949** Donald Hebb[2] proposed a learning scheme for updating neuron's connections.

Lack of efficient learning schemes and limited computational resources slowed the neural network development effort until the 1980s.

**1982-1984** John Hopfield[3, 4] introduced recurrent neural network architecture for associative memories.

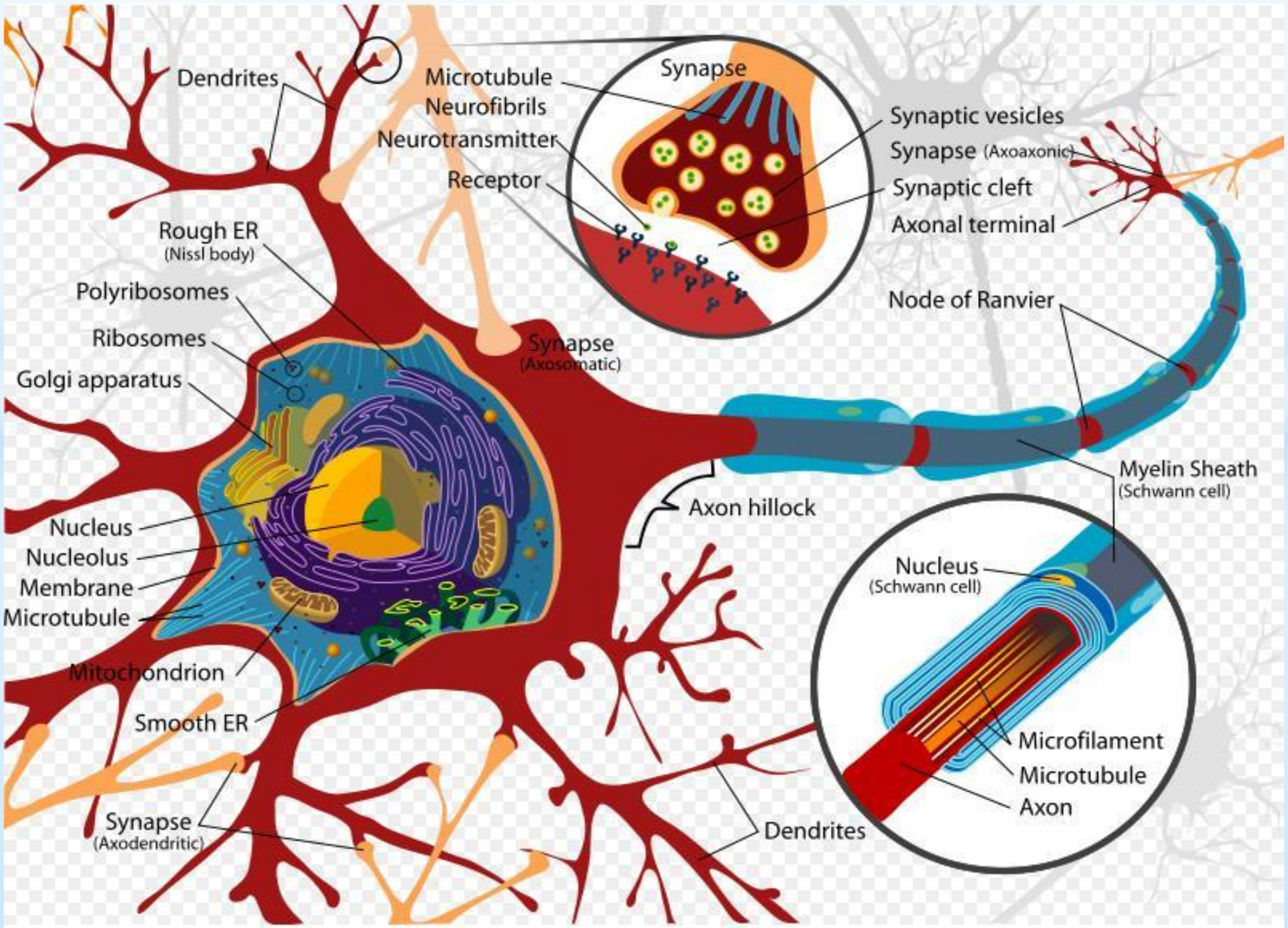
**1986** The Back-Propagation learning algorithm for Multi-Layer Perceptron was rediscovered and the whole field took off again.



# Outline

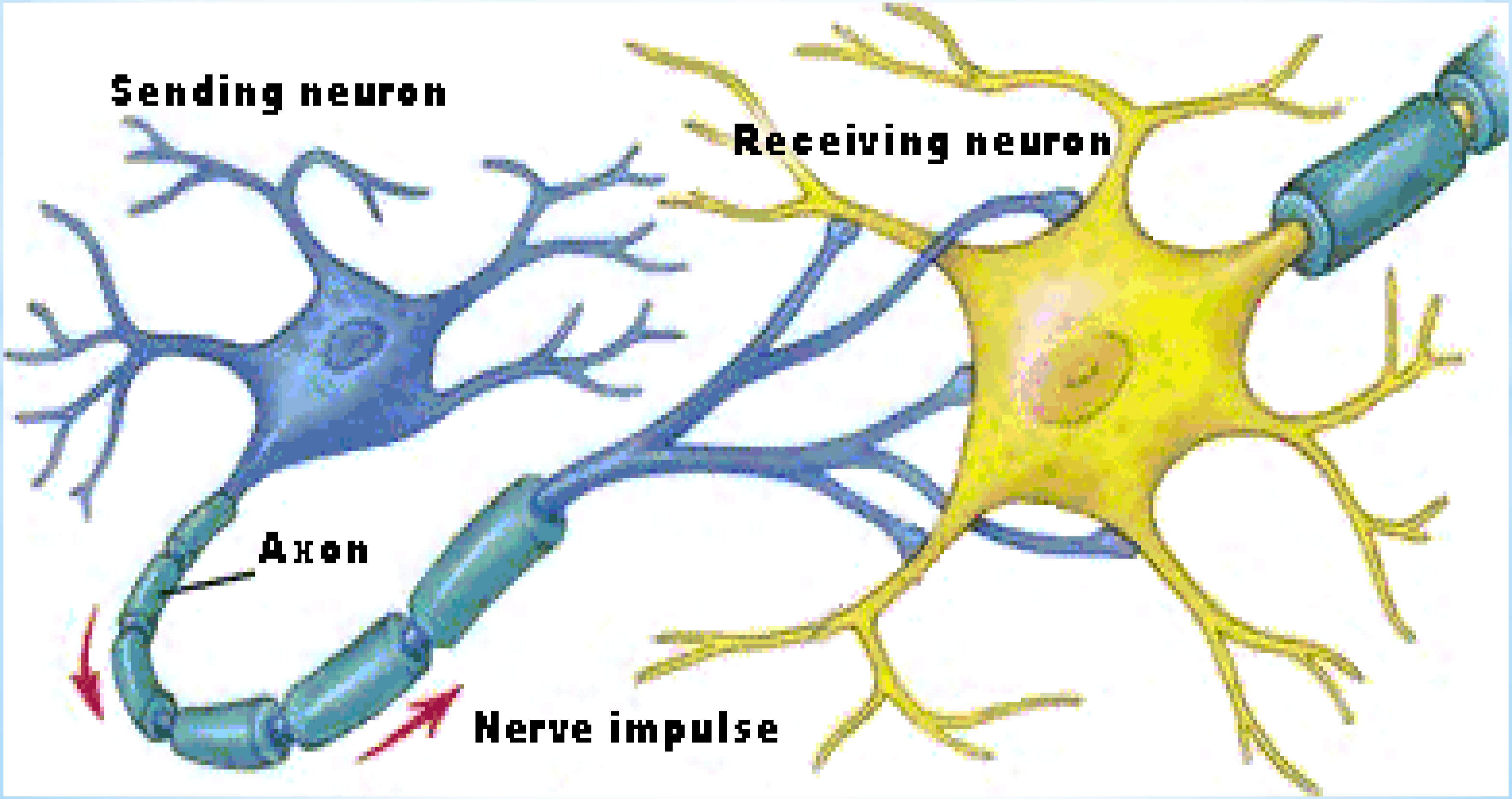
- Definitions and Background
- **Biological and Artificial Neuron**
- Activation Functions and ANN layers
- Types of ANN
- Training of ANN
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- Applications of ANN

# Biological Neuron



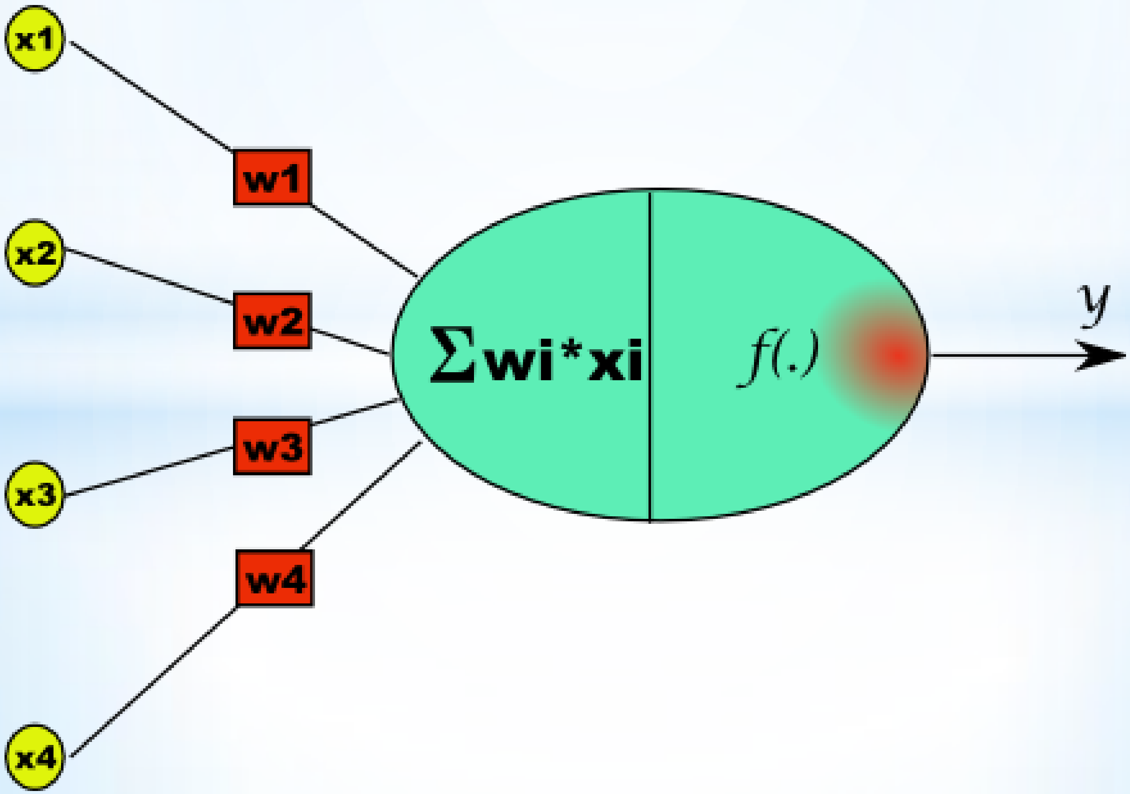

http://toxipedia.org/download/attachments/10435/neurotoxin1.jpg

# Biological Neuron




http://www.optimaltrader.net/neural_network.htm

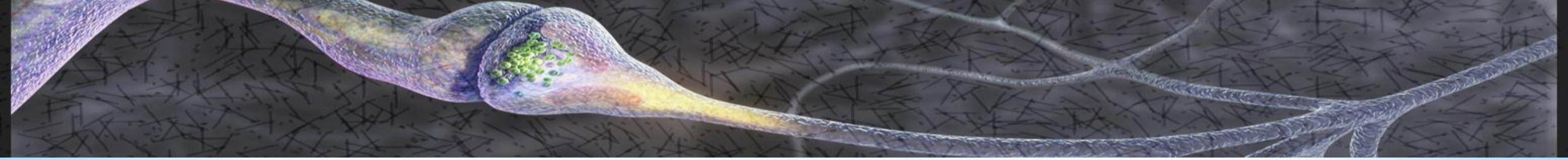
# Artificial Neuron



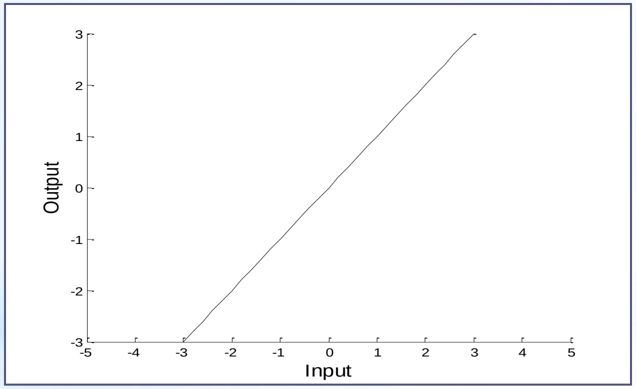

http://www.realintelligence.net/tut_perceptron

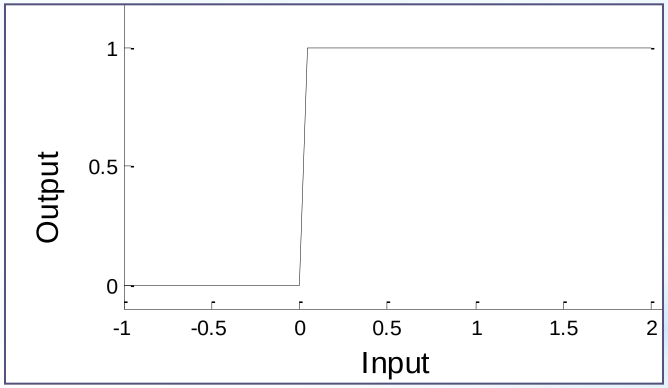
# Outline

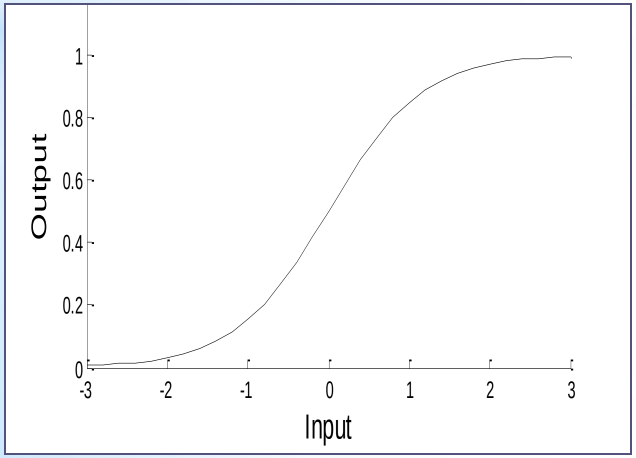
- Definitions and Background
- Biological and Artificial Neuron
- **Activation Functions and ANN layers**
- Types of ANN
- Training of ANN
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- Applications of ANN

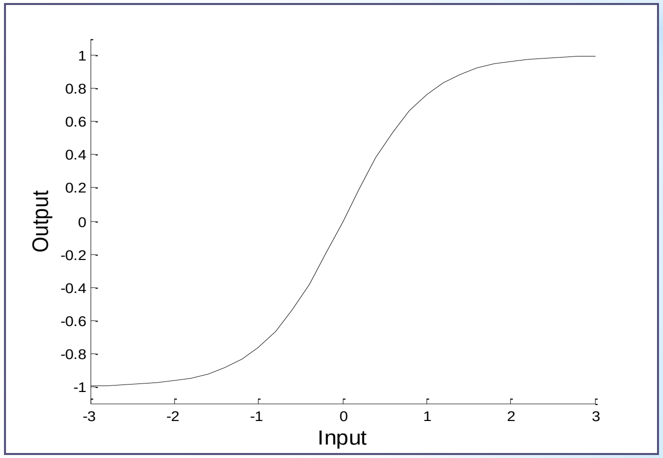


# Activation Functions

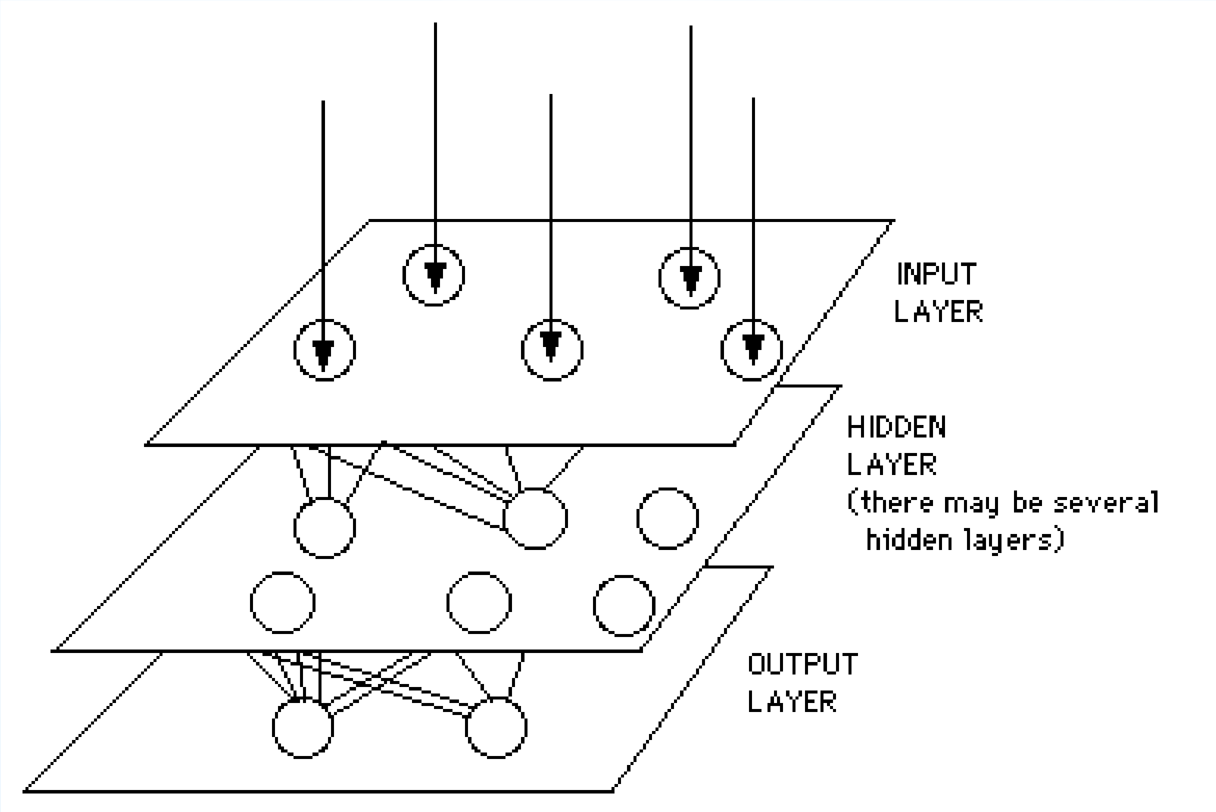


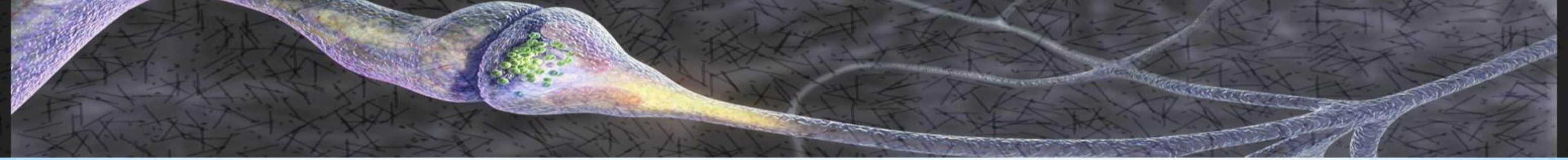



# Layer arrangements in ANN

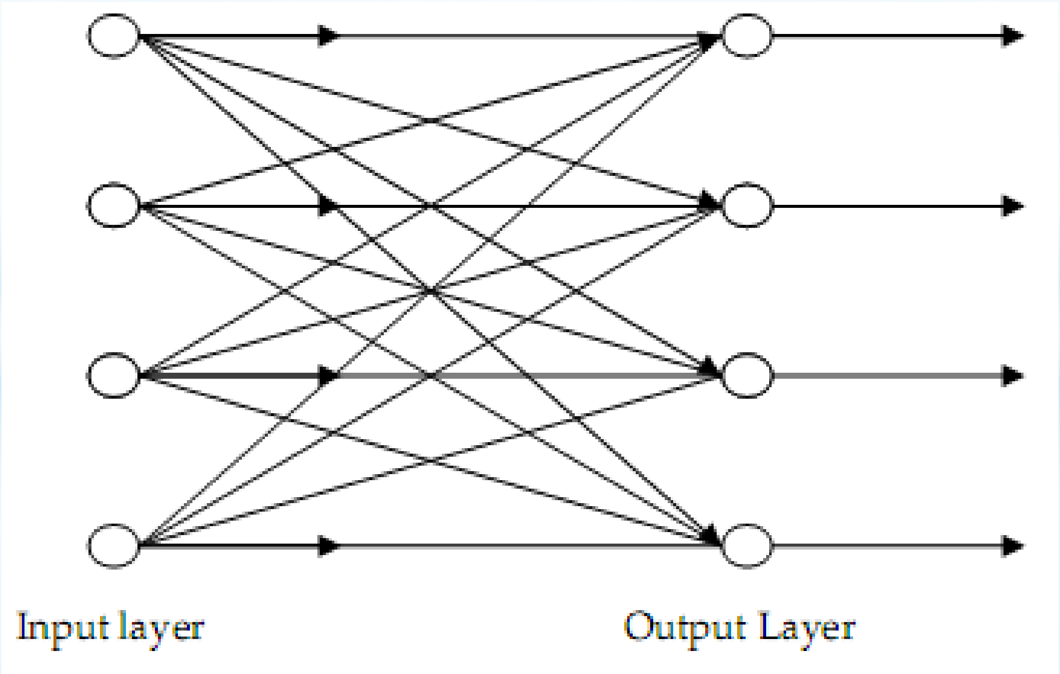




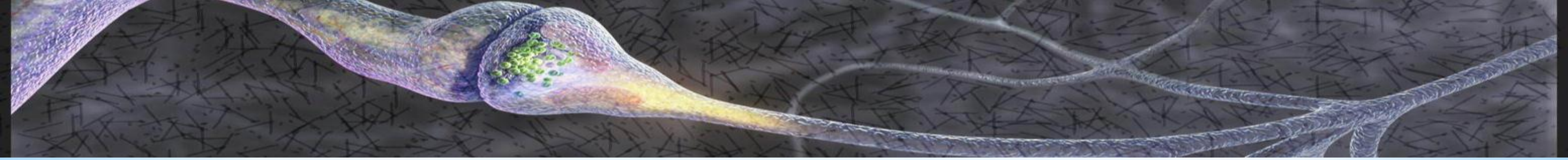
# Outline

- Definitions and Background
- Biological and Artificial Neuron
- Activation Functions and ANN layers
- **Types of ANN**
- Training of ANN
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- Applications of ANN

# Single Layer Feed Forward Network

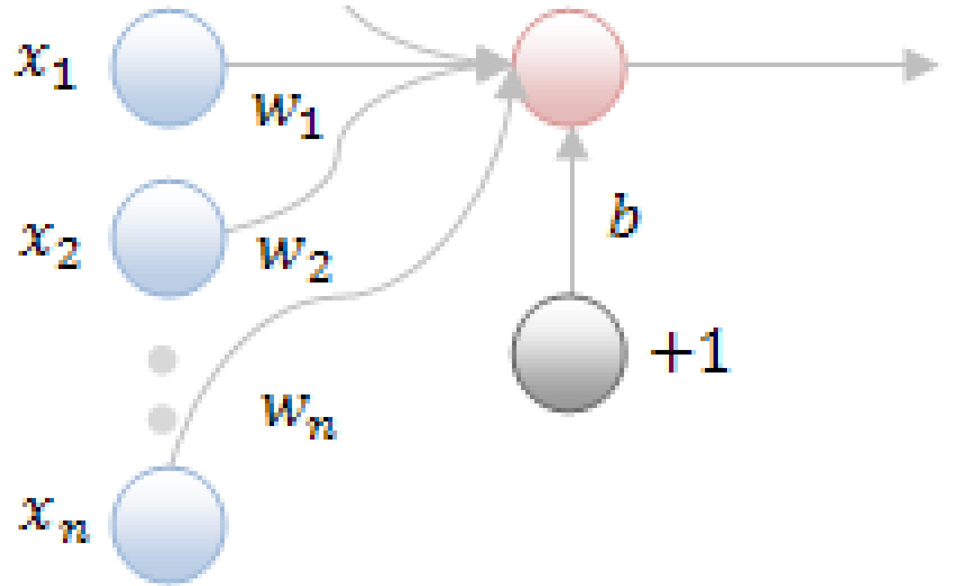



[10]



# The Perceptron

$$Net = \sum_{i=0}^{n} x_i w_i + b \qquad f(x) = \left\{ \frac{x \le 0 \rightarrow 0}{x > 0 \rightarrow 1} \right\}$$

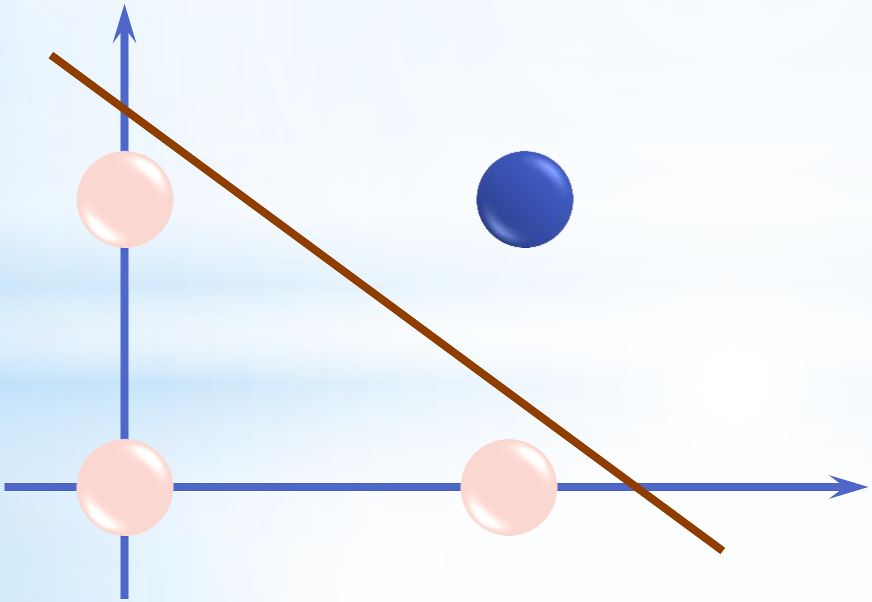



Single Layer Perceptron

http://www.generation5.org/content/1999/perceptron.asp

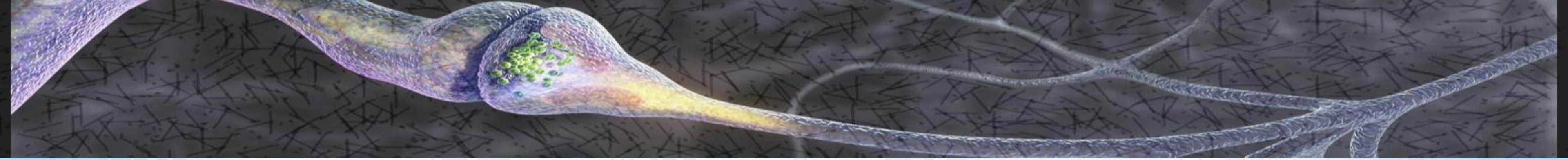
# The Single Layer Perceptron

## AND Gate



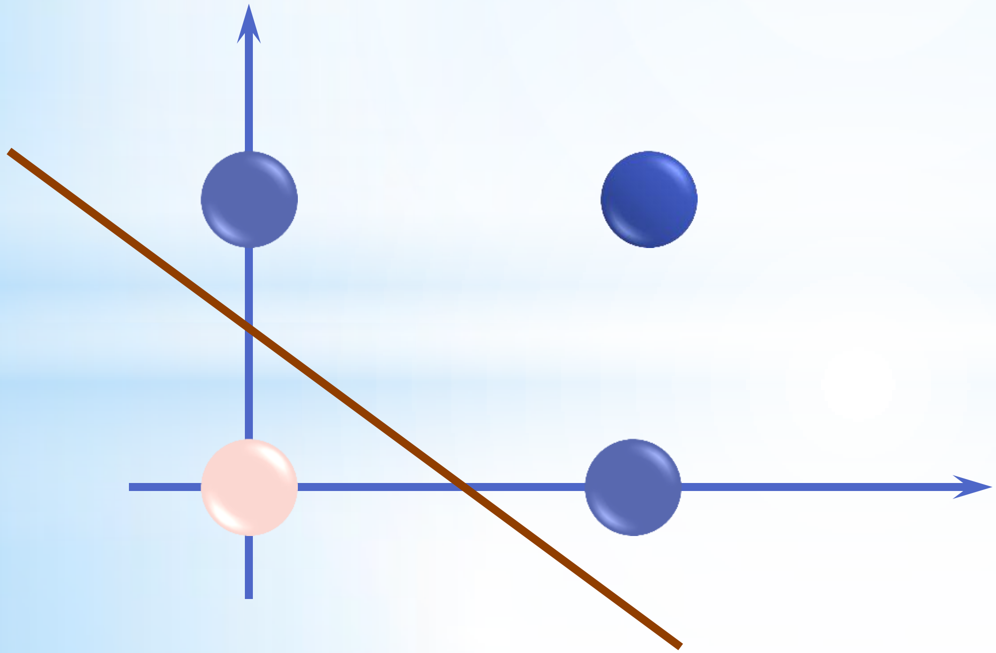
| $X_1$ | $X_2$ | y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

$$net = \sum_{i=0}^{2} w_i x_i + b$$

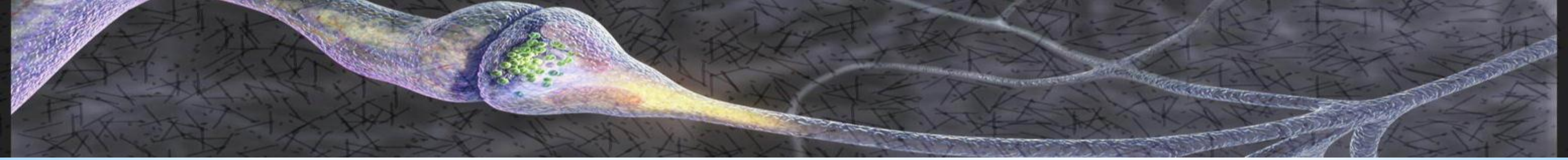


# The Single Layer Perceptron

## OR Gate

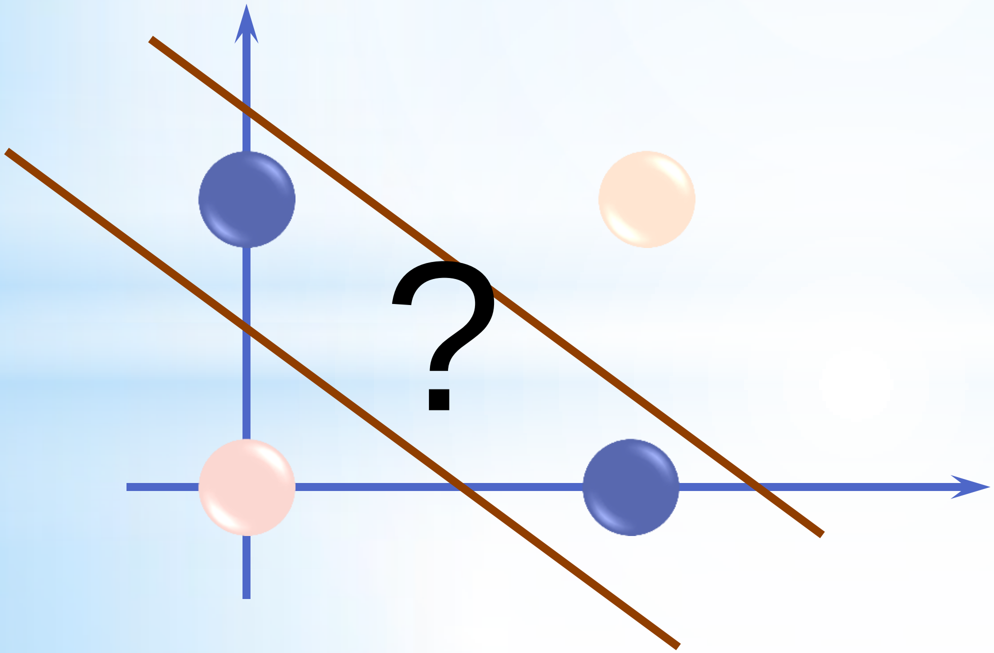


| $X_1$ | $X_2$ | y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |



# The Multi Layer Perceptron

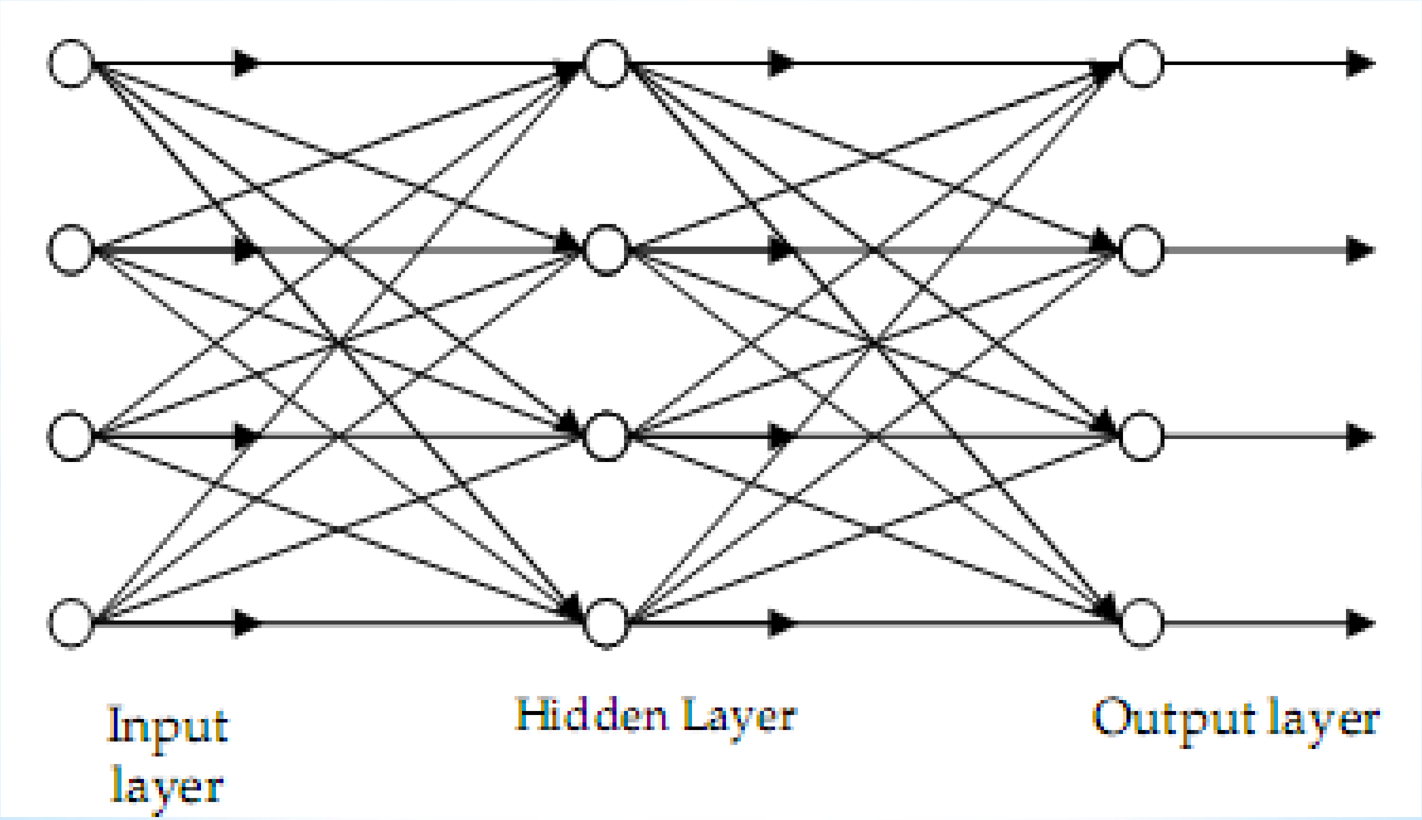
## XOR Gate




| $X_1$ | $X_2$ | y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

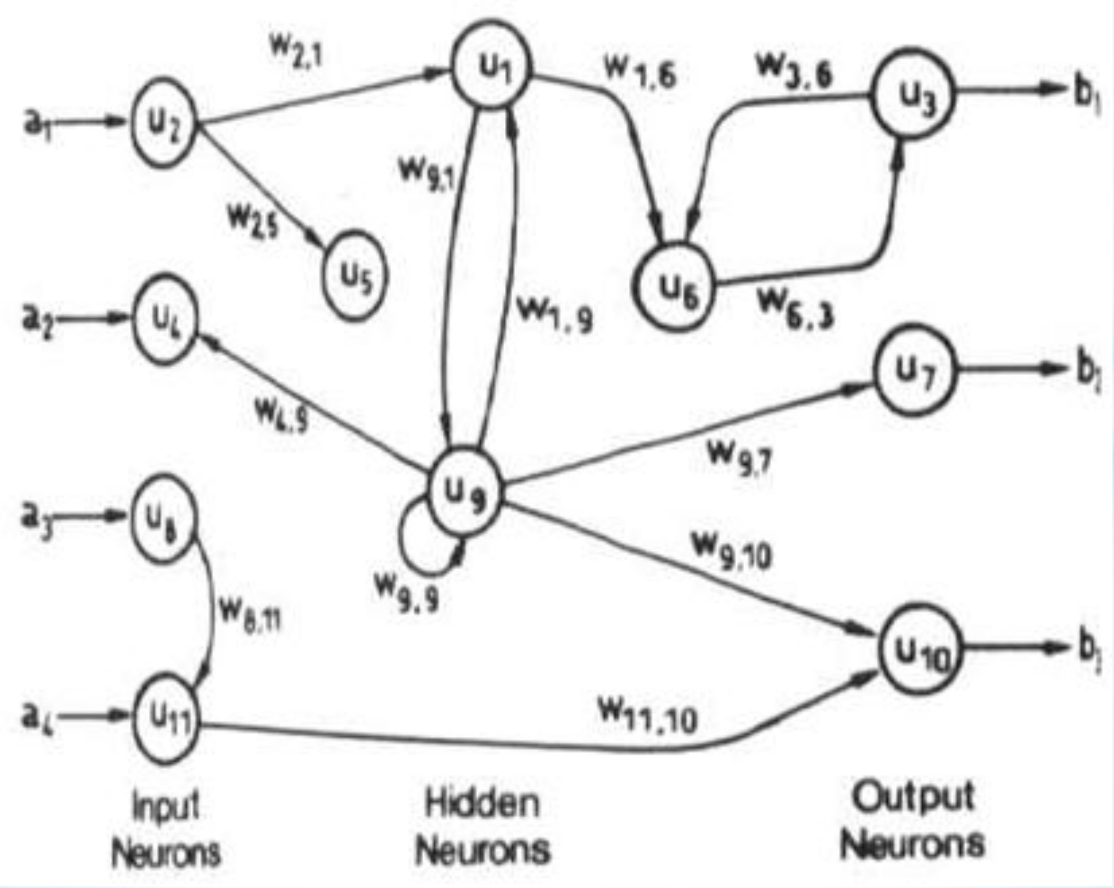
Non-Linearly Separable Data

# Multi Layer Feed Forward Network

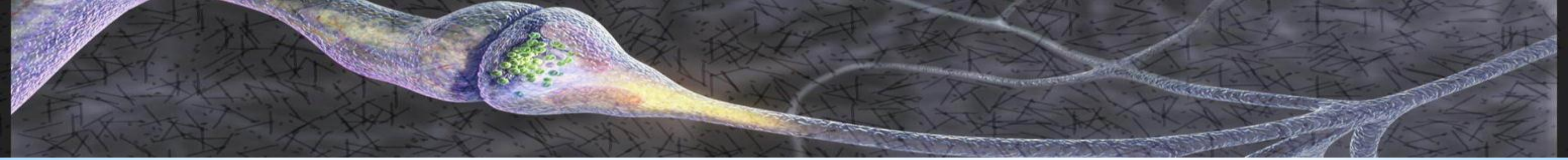



[10]

# Recurrent Networks




[10]



# Outline

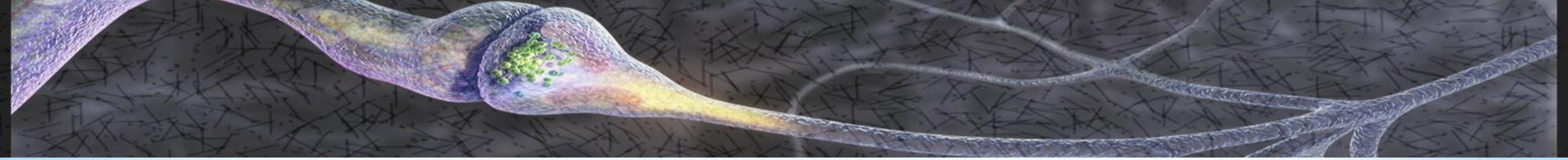
- Definitions and Background
- Biological and Artificial Neuron
- Activation Functions and ANN layers
- Types of ANN
- **Training of ANN**
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- Applications of ANN

# Training of ANN

- Supervised Training
- Unsupervised or Adaptive Training
- Reinforcement learning (i.e. learning with limited feedback)

# Supervised Training

- The inputs and the outputs  are provided.
- The network compares its resulting outputs against the desired outputs.
- Errors are used to adjust the weights.
- Weights adjustment continue until the performance index hits specific limit.



# Unsupervised Training

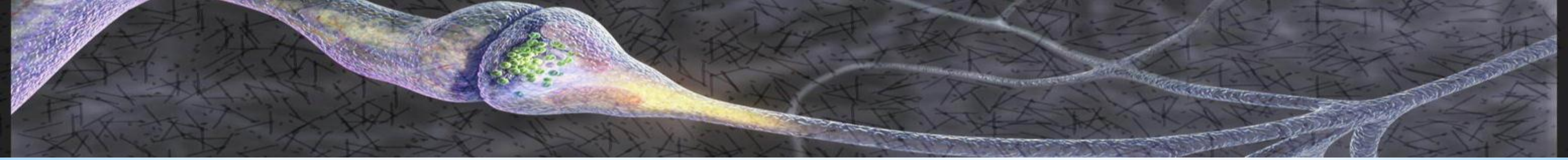
- Only the inputs are provided.
- Self-Organization and adaption.
- Self-performance monitoring.

# Outline

- Definitions and Background
- Biological and Artificial Neuron
- Activation Functions and ANN layers
- Types of ANN
- Training of ANN
- **Learning Rates and Learning Algorithms**
- Back propagation FFANN
- Applications of ANN

# Learning Rate

- Slower rate means more time for training.
- Faster rate; no fine discriminations.
- Learning rate $0 < \eta \leq 1$

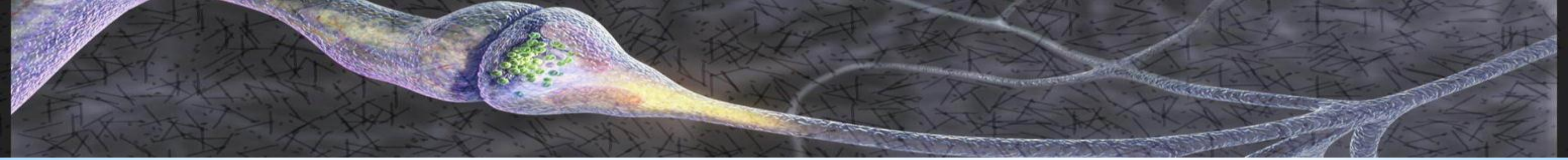


# Learning Algorithms

**Hebb's Rule**: If a neuron receives an input from another neuron and if both are highly active (same sign), the weight between the two neurons should be strengthened.

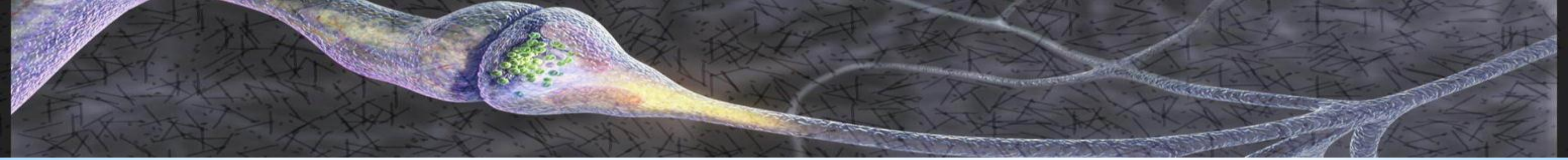
# Learning Algorithms

**<u>Hopfield Law:</u>** If the desired output and the input are both active or both inactive, increment the connection weight by the learning rate, otherwise decrement the weight by the learning rate.



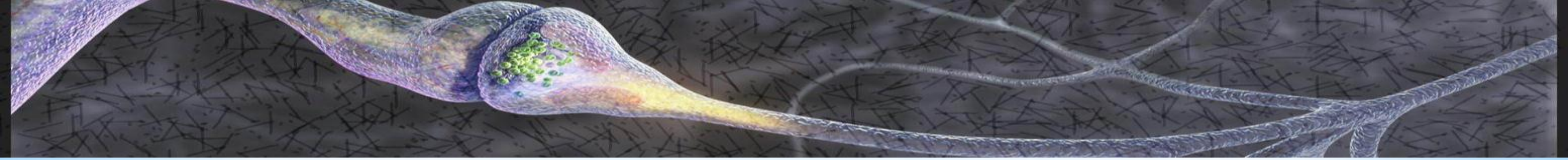
# Learning Algorithms

**<u>The Delta Rule:</u>** modifying the strengths of the input connections to reduce the difference (the delta) between the desired output value and the actual output of a processing element.



# Learning Algorithms

**The Gradient Descent Rule:** the derivative of the activation function is used to modify the delta error.

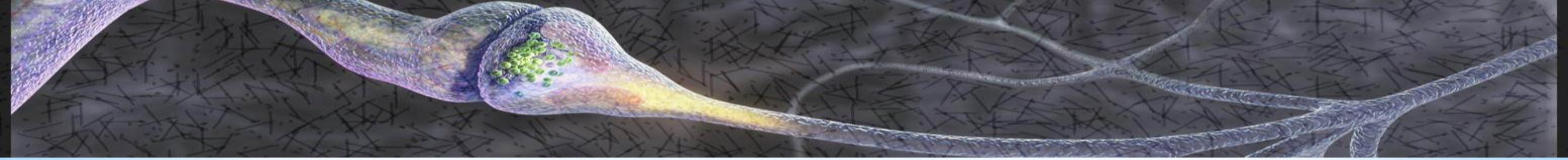


# Learning Algorithms

**<u>Kohonen's Law:</u>** the processing elements compete for the opportunity to learn or update their weights.

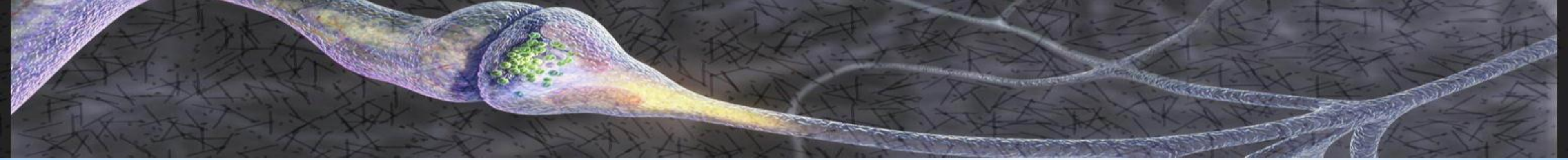
# Learning Algorithms

**<u>Levenberg–Marquardt algorithm:</u>** provides a solution to the problem of minimizing a nonlinear function over a space of parameters of the function. It is more robust than the Gauss-Newton Algorithm



# Outline

- Definitions and Background
- Biological and Artificial Neuron
- Activation Functions and ANN layers
- Types of ANN
- Training of ANN
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- Applications of ANN



# Back Propagation Learning

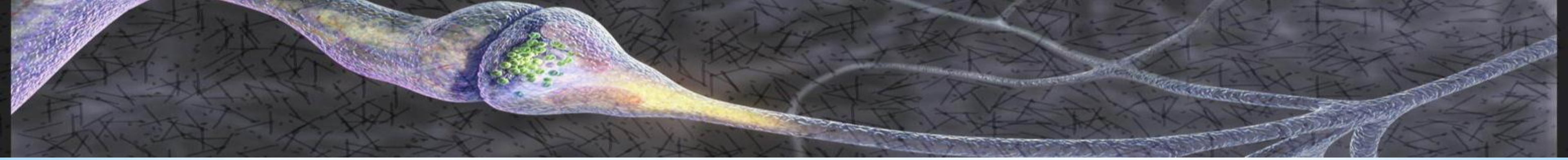
Training process is achieved as follows.

1. Initial values of weights are assumed
2. The network outputs are calculated for the input set
3. The error with respect to the desired output is calculated
4. The error is back propagated through the network updating the output layer's weights then the hidden layer's weights
5. The process is repeated until MSE is minimized

# Back Propagation Learning

Mathematically

- Define the Error as $$E = \sum_{n=1,\dots,T} \left\| d(n) - y(n) \right\|^2 = \sum_{n=1,\dots,T} E(n)$$
- To minimize the error, the weights should be changed along the gradient of the error wrt weights $$\frac{\partial E}{\partial w_{ij}^{m}} = \sum_{t=1,\dots,T} \frac{\partial E(n)}{\partial w_{ij}^{m}}$$
- So, $$new\ w_{ij}^{m} = w_{ij}^{m} - \gamma \frac{\partial E}{\partial w_{ij}^{m}}$$



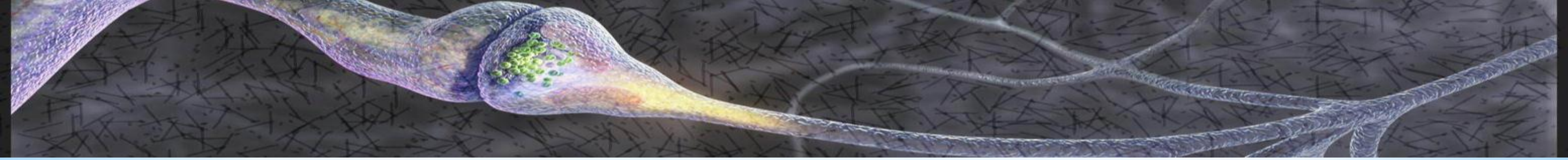
# Back Propagation Learning

Mathematically

- Define the Error as

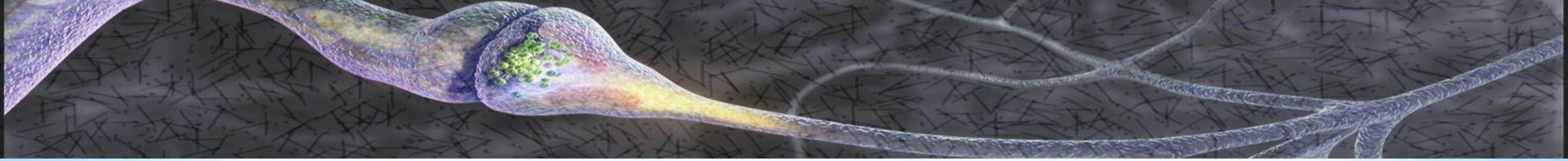
$$new\ w_{ij}^{m-1} = w_{ij}^{m-1} + \gamma \sum_{t=1}^{T} \delta_i^{m}(n) x_j^{m-1}(n)$$

- $\delta_i^{m}(n)$ Represents the error propagation term



# Outline

- Definitions and Background
- Biological and Artificial Neuron
- Activation Functions and ANN layers
- Types of ANN
- Training of ANN
- Learning Rates and Learning Algorithms
- Back propagation FFANN
- **Applications of ANN**

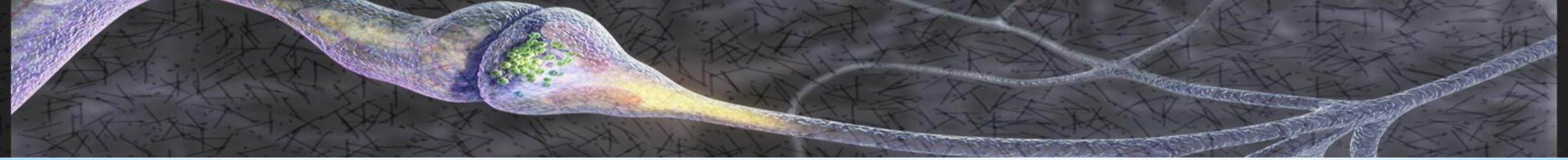


# General Applications

- Language Processing
- Character Recognition
- Image compression
- Pattern Recognition
- Signal Processing
- Financial
- Servo Control

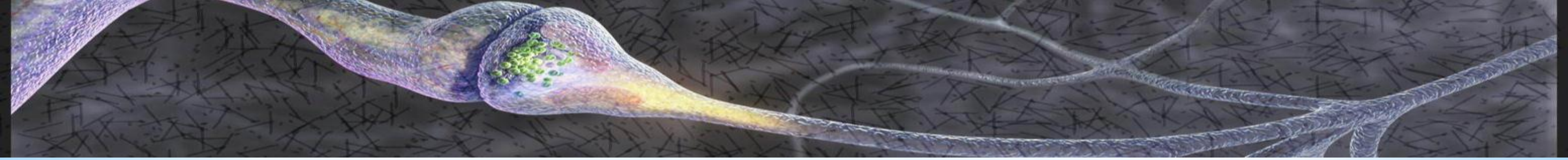
# Power Systems Applications

- Load Forecasting
- Fault Diagnosis\Fault Location
- Economic Dispatch
- Automatic Generation Control.
- Power System Stabilizer
- Harmonic Source Monitoring
- Power Flow and Load Modeling



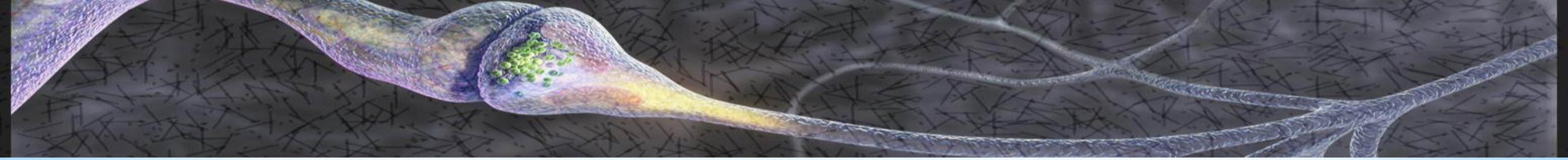
# Power Systems Applications

- Load Forecasting
- Fault Diagnosis\Fault Location
- Economic Dispatch
- Automatic Generation Control.
- Power System Stabilizer
- Harmonic Source Monitoring
- Power Flow and Load Modeling



# References


1. W. S. McCulloch and W. H. Pitts,“ *A logical calculus of the ideas imminent in nervous activity*”, *Bull. Math. Biophy*. Vol. 5, 1943, pp. 115-133.
2. D. O. Hebb,“ *The organization of behavior, a neuropsychological theory*”, John Wiley, New York, 1949.
3. J. J. Hopfield,“ *Neural networks and physical systems with emergent collective computational abilities*”, *Proc. of Nat’s Acad. Science*, Vol. 79, 1982, pp. 2554-2558.
4. J. Hopfield,“ *Neurons with graded response have collective computational properties like those of two state neurons*”, *Proc. of Nat’l Acad. Science*, Vol. 81, 1984, pp. 3088-3092.
5. J. McClelland and D. E. Rumelhart, “*Parallel distributed processing*”, MIT press and the PDP Research Group, Cambridge, 1986.
6. A. Hariri,“ *An Adaptive fuzzy logic power system stabilizer*”, Ph. D. Dissertation, University of Calgary, AB, 1997.
7. M. Norgaard et al.,“ Neural Networks for modeling and control of dynamic systems”, 3rd. edition, 2003, Springer-Verlag London Limited.



# References


8. T. Kohonen;“ *Self-organizing map*”, *IEEE Proc.*, 1990, Vol. 78, No.9,pp 1464-1480
9. G. A. Carpenter and S. Grossberg; “*The ART of adaptive pattern recognition by a self-organizing neural network*”, IEEE Computer, March 1988, pp.77-88.
10. M. Y. Soliman; “*Dynamic Analysis of Fuel Cell/Microturbine Generation Scheme with Neural Network Control for Peak Power Shaver Application”,* Ph.D. dissertation, University at Buffalo, February 2007.
11. Hosny, A. A.; and Safiuddin, M. *ANN-based protection system for Controllable Series-Compensated transmission lines*. Power Systems Conference and Exposition, 2009. PES '09. IEEE/PES, March 15, 2009.
12. Cover picture is from http://1.bp.blogspot.com/_wbp_krRPdF0/TIomUBzg4GI/AAAAAAAAAIs/FOxBQ2ebrnw/s1600/Synapse-Structure.jpg


COFFEE TIME !
Please sit back and enjoy your coffee.
motifake.com