
Programming

A scholar in hard and soft sciences has to have good skills in **mathematics**, and **statistics** (and **econometrics**)

With the diffusion of computer simulation, the same is true about **programming knowledge**

Using what kind of tools? High level ones, as most important packages for calculus, statistics, econometrics, simulation ..., but having also a close look to the modern foundations of programming, via an object oriented language

Programming methodologies

- imperative programming https://en.wikipedia.org/wiki/Imperative_programming as sequences of orders to be executed in a rigid way
 - with the object-oriented paradigm, we use objects and methods as metaphorical representations of the actual world
- declarative programming http://en.wikipedia.org/wiki/Declarative_programming
 - definitions and “engines” to use them
- *soft computing*
 - neural networks http://en.wikipedia.org/wiki/Neural_network
 - genetic algorithms http://en.wikipedia.org/wiki/Genetic_algorithms
 - classifier systems http://en.wikipedia.org/wiki/Learning_classifier_system

Imperative programming, calculating a factorial (in **Python**)

```
n=5
```

```
f=1.
```

```
for i in range(1, 6):
```

```
    f=f*i
```

```
print (f)
```

Declarative programming, calculating a factorial (in **Maxima**,
<http://maxima.sourceforge.net>)

```
fac(n) := if n = 0 then 1 else n*fac(n-1)
```

Declarative programming, calculating a factorial (in **Python**)

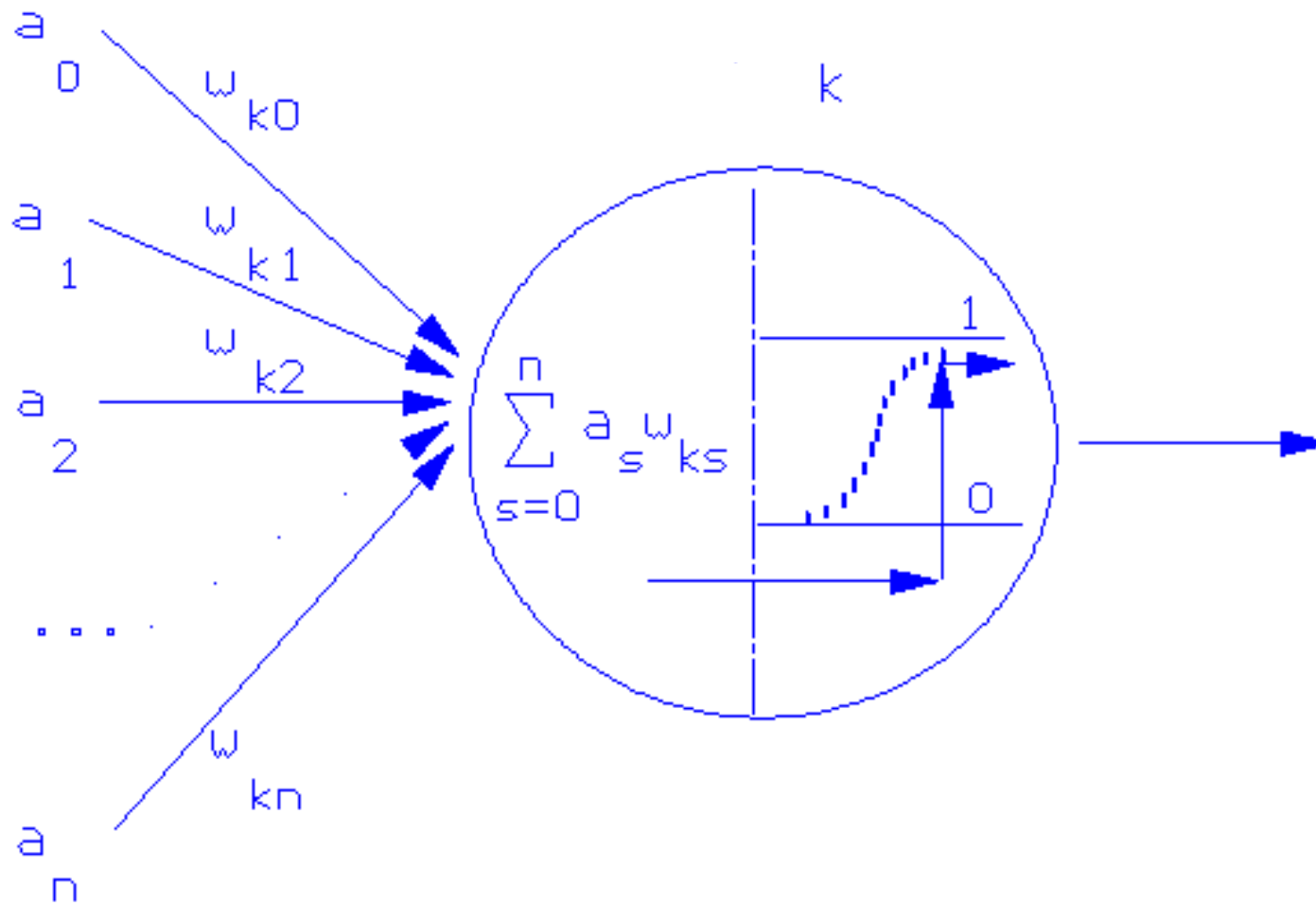
```
def fac(n):  
    if n == 0: return 1  
    return n*fac(n-1)
```

About recursion have a look at

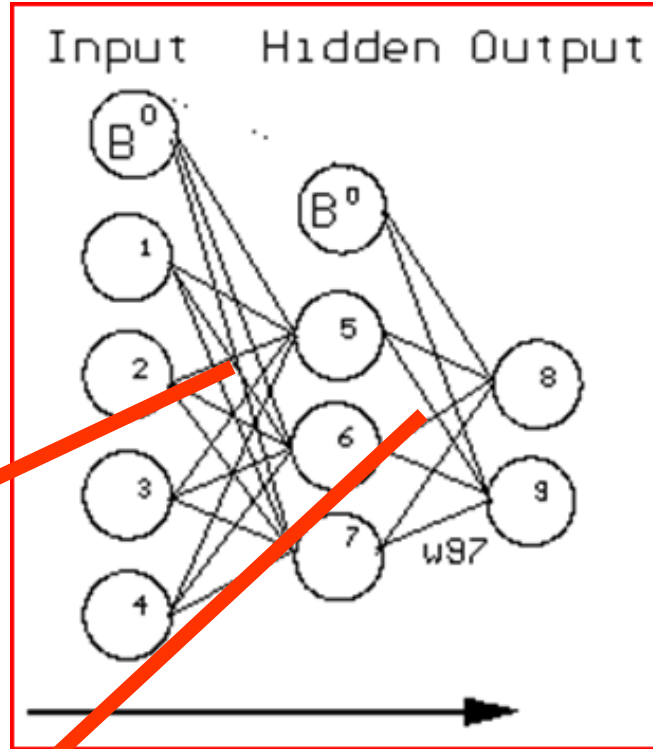
<http://www.shodor.org/interactivate/activities/Recursion/>

Artificial neural networks

A neuron or node of the network



The network




$$A = \begin{array}{|c|c|c|} \hline w50 & w60 & w70 \\ \hline w51 & w61 & w71 \\ \hline w52 & w62 & w72 \\ \hline w53 & w63 & w73 \\ \hline w54 & w64 & w74 \\ \hline \end{array}$$

$$B = \begin{array}{|c|c|} \hline w80 & w90 \\ \hline w85 & w95 \\ \hline w86 & w96 \\ \hline w87 & w97 \\ \hline \end{array}$$

With $f(x) = 1/(1 + e^{-x})$ we can write:
 $y = f(B [1, f(A [1, x])])$

where $y = [y_8 \ y_9]'$ is the output vector and

$x = [x_1 \ x_2 \ x_3 \ x_4]'$ is the input vector; the constant 1 is added as first element of the vector as input values for the so called bias knots or neurons

$$y = f(B f(A x))$$


Considering the error $E = \sum (t_k - y_k)^2$ we can
calculate the correction
of each w_{ij} parameter
following a set of (x,y) statistics

To read more about Artificial Neural Networks, from my home page:

A recent thesis (Gabriele D'Acunto) at

<http://terna.to.it/tesi.html>

<https://learningtensorflow.com>

<http://tflearn.org>

A quite old book with a special chapter:

Beltratti, A., Margarita S., Terna P. (1996a), *Neural Networks for Economic and Financial Modelling*, ITCP, London. http://terna.to.it/deposito/ct_1996.pdf