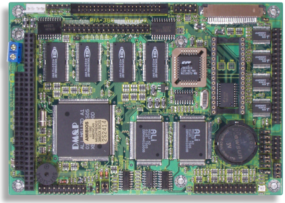
The MINT neural network library

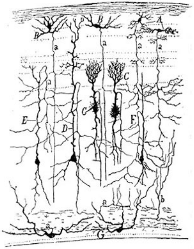
A quick introduction

Stefano Ghirlanda

June 7th, 2011

Why neural networks





- Dominant paradigm for understanding intelligence: the digital computer
- Natural intelligence is done by brains
- Neural networks try to capture the “computational style” of brains

Neural networks in a nutshell

- Made of “nodes” (neurons) that can be in different states of activity (on-off, between 0 and 1, etc.)
- Nodes influence each other through “connection weights” (synapses) which can be positive/negative (excitatory/inhibitory), strong or weak
- Some nodes are “input” (sensors), others “ouptut” (e.g., motor neurons)

Neural networks in a nutshell

- Made of "nodes" (neurons) that can be in different states of activity (on-off, between 0 and 1, etc.)
- Nodes influence each other through "connection weights" (synapses) which can be positive/negative (excitatory/inhibitory), strong or weak
- Some nodes are "input" (sensors), others "ouptut" (e.g., motor neurons)
- Node activity changes in time reflecting input values and characteristics of the network
- Connections may change, too ("learning")

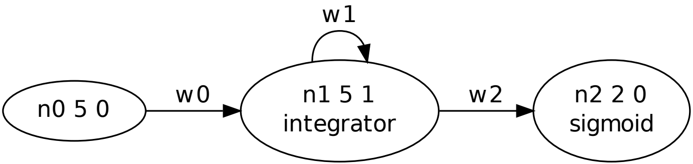
Neural networks in a nutshell

- Made of “nodes” (neurons) that can be in different states of activity (on-off, between 0 and 1, etc.)
- Nodes influence each other through “connection weights” (synapses) which can be positive/negative (excitatory/inhibitory), strong or weak
- Some nodes are “input” (sensors), others “ouptut” (e.g., motor neurons)
- Node activity changes in time reflecting input values and characteristics of the network
- Connections may change, too (“learning”)
- **Goal: understanding how all this generates behavior!**

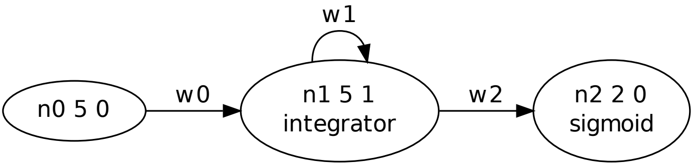
Why MINT

- Computer simulation is crucial in neural network modeling
- Problems of existing software:
  - Engineering view of neural networks
  - Not general
  - Too low level for my (our) purpose
  - Makes things complicated
  - Costs money

Example

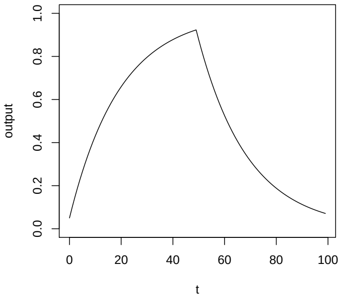


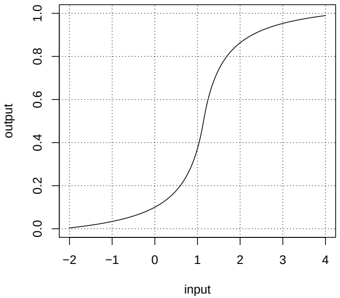
Example



```
network 3 3
nodes 5 0
nodes 5 1 update  integrator  10 .25
nodes 2 0 update sigmoid .1  1
weights  5 5 0 0 1
weights  5 5 0 1 1
weights  5 2 0 1 2
```

“Integrator” and “sigmoid”





Code

Creating the network:

```
struct mint_network *net;
file = fopen( "recnet.arc", "r" );
net = mint_network_load( file );
fclose( file );
```

Code

Creating the network:

```
struct mint_network *net;
file = fopen( "recnet.arc", "r" );
net = mint_network_load( file );
fclose( file );
```

Setting the weights:

```
mint_weights_set_uniform( mint_network_weights(net,0), .5 );
mint_weights_set_random( mint_network_weights(net,1), -1, .25 );
mint_weights_set_uniform( mint_network_weights(net,2), .5 );
```

Code

Creating the network:

```
struct mint_network *net;
file = fopen( ”recnet.arc”, ”r” );
net = mint_network_load( file );
fclose( file );
```

Setting the weights:

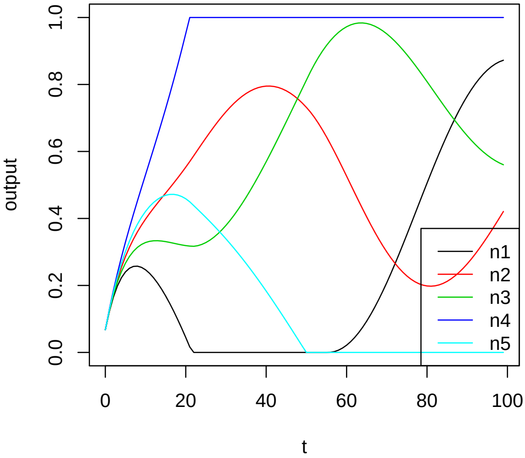
```
mint_weights_set_uniform( mint_network_weights(net,0), .5 );
mint_weights_set_random( mint_network_weights(net,1), -1, .25 );
mint_weights_set_uniform( mint_network_weights(net,2), .5 );
```

Running the network:

```
for( t=0; t<100; t++ ) {
   khepera_input( net );
   mint_network_nupdate( net );
   khepera_output( net );
}
```

Results

“Brain” activity in response to constant input:



Analysis

...

```
weights 5 5 0 1 1
-0.190695 -0.890721 0.020717 -0.117820 -0.766349
0.034132 -0.208735 -0.952190 0.112449 -0.426193
-0.304586 0.497512 -0.515037 -0.373985 -0.938971
-0.425993 0.370224 -0.485271 -0.051057 -0.345650
0.030423 -0.221058 -0.309130 -0.451573 -0.253704
```

...

What next

- Documentation
- Testing
- Features:
  - Learning
  - Analysis / Visualization
  - Interfacing with Khepera
  - . . .