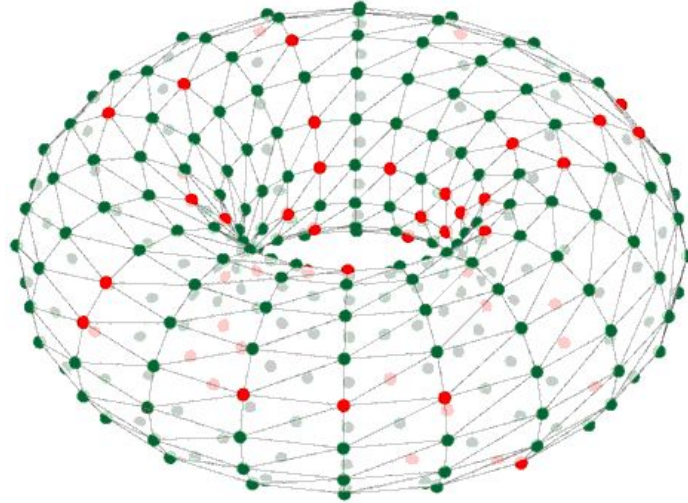


SpiNNaker Tutorial



European Research Council
Established by the European Commission



Human Brain Project

EPSRC



Accessing SpiNNaker

Accessing SpiNNaker via Jupyter

<https://spinn-20.cs.man.ac.uk/>

User Guide

Sign in with EBRAINS Credentials

Sign in with HBP Credentials

Accessing SpiNNaker via Jupyter

User Guide

[Jupyter Notebook Interface](#)

[JupyterLab Interface](#)

Accessing SpiNNaker via Jupyter

The image shows the Jupyter Launcher interface. On the left is a file browser showing a directory structure with folders like '01.RunningPyNNSimulations', '02.LiveInputAndOutput', '03.NeuroboticsPlatform', 'application_generated_data...', 'Documents', 'drive', 'nginx', 'nrpStorage', 'reports', 'sPyNNaker', 'sPyNNakerGit', 'work', and files 'SpaceInvaders.ipynb' and 'SynfireExample.ipynb'. The main area is titled 'Launcher' and contains three sections: 'Notebook', 'Console', and 'Other'. Each section has four icons representing different environments: Python 3, Python 2, sPyNNaker, and sPyNNakerGit. The 'Other' section includes icons for Terminal, Text File, Markdown File, and Show Contextual Help.

File Edit View Run Kernel Git Tabs Settings Help

Launcher

Notebook

Python 3 Python 2 sPyNNaker sPyNNakerGit

Console

Python 3 Python 2 sPyNNaker sPyNNakerGit

Other

Terminal Text File Markdown File Show Contextual Help

0 \$ _ 0

Launcher

Jupyter Folders

The image shows the Jupyter Launcher interface. On the left is a file explorer sidebar with a list of folders and files. On the right is the main launcher area with three sections: Notebook, Console, and Other.

File Explorer (Left):

- 01.RunningPyNNSimulations
- 02.LiveInputAndOutput
- 03.NeuroboticsPlatform
- application_generated_data...
- Documents
- drive
- nginx
- nrpStorage
- reports
- sPyNNaker
- sPyNNakerGit
- work
- SpaceInvaders.ipynb
- SynfireExample.ipynb

Launcher (Right):

- Notebook:** Python 3, Python 2, sPyNNaker, sPyNNakerGit
- Console:** Python 3, Python 2, sPyNNaker, sPyNNakerGit
- Other:** Terminal, Text File, Markdown File, Show Contextual Help

Red arrows point from the file explorer to the 'drive' folder and the 'work' folder. A red bracket groups the folders 01 through 03.

EBRAINS Drive

The image shows a screenshot of the EBRAINS Drive Launcher application. The interface is divided into a left sidebar and a main content area.

Left Sidebar:

- Menu: File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help
- File Explorer: Shows the path `/ drive /` and a list of folders: My Libraries, Shared with all, Shared with groups, and Shared with me.
- Toolbars: A vertical toolbar on the left contains icons for home, search, refresh, settings, and other functions.

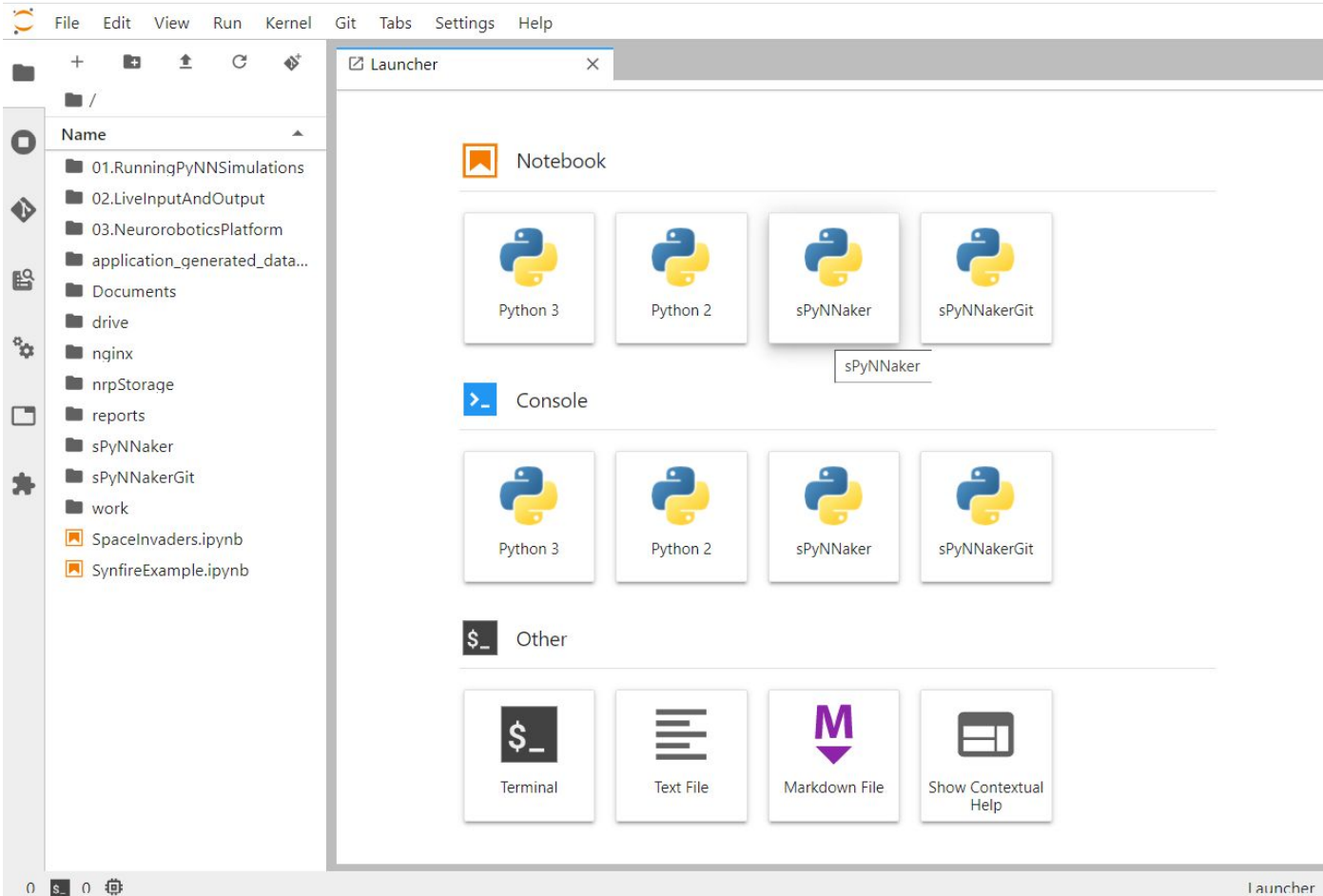
Main Content Area (Launcher):

- Section: **drive**
- Category: **Notebook** (indicated by a notebook icon)
- Options: Four notebook icons are displayed, labeled Python 3, Python 2, sPyNNaker, and sPyNNakerGit.
- Category: **Console** (indicated by a terminal icon)
- Options: Four console icons are displayed, labeled Python 3, Python 2, sPyNNaker, and sPyNNakerGit.
- Category: **Other** (indicated by a terminal icon)
- Options: Four other icons are displayed: Terminal, Text File, Markdown File, and Show Contextual Help.

Bottom Status Bar:

- Left: `1 $ _ 0` and a gear icon.
- Right: Launcher

Running sPyNNaker



Running sPyNNaker

The screenshot displays a Jupyter Notebook environment. The top menu bar includes 'File', 'Edit', 'View', 'Run', 'Kernel', 'Git', 'Tabs', 'Settings', and 'Help'. The left sidebar shows a file explorer with a tree view of the current directory. The main area is a code editor for 'Untitled.ipynb', showing a single code cell with a cursor at the beginning of a line. The status bar at the bottom indicates 'Mode: Edit', 'Ln 1, Col 1', and 'Untitled.ipynb'.

Changing Kernel

The screenshot displays the JupyterLab interface. On the left, a file browser shows a directory structure with files like 'Untitled.ipynb' selected. The main area shows a Jupyter Notebook with a terminal at the bottom. The 'Kernel' menu is open, listing options such as 'Interrupt Kernel', 'Restart Kernel...', and 'Change Kernel...'. The 'Change Kernel...' option is highlighted. The status bar at the bottom indicates 'Mode: Command' and 'Ln 1, Col 1'.

Kernel

- Interrupt Kernel 1, 1
- Restart Kernel... 0, 0
- Restart Kernel and Clear All Outputs...
- Restart Kernel and Run up to Selected Cell...
- Restart Kernel and Run All Cells...
- Shut Down Kernel
- Shut Down All Kernels...
- Change Kernel...

0 \$ 1 sPyNNaker | Idle Mode: Command Ln 1, Col 1 Untitled.ipynb

Close Notebooks

The screenshot shows the JupyterLab application window. The 'File' menu is open, displaying various options. The option 'Close and Shutdown Notebook' is highlighted, with the keyboard shortcut 'Ctrl+Shift+Q' shown to its right. Other visible options include 'New', 'New Launcher', 'Open from Path...', 'New View for Notebook', 'New Console for Notebook', 'Close Tab', 'Close All Tabs', 'Save Notebook', 'Save Notebook As...', 'Save All', 'Reload Notebook from Disk', 'Revert Notebook to Checkpoint', 'Rename Notebook...', 'Download', 'Export Notebook As...', 'Print...', 'Hub Control Panel', and 'Log Out'. The background shows a notebook interface with a toolbar and a status bar at the bottom.

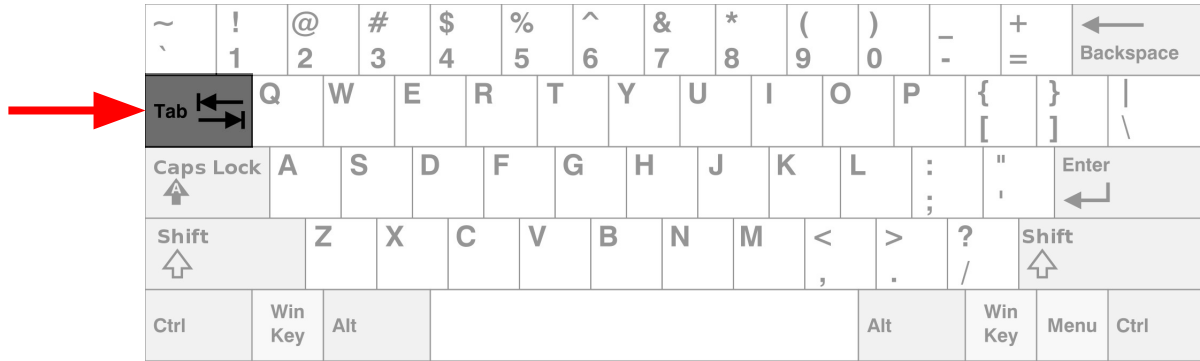
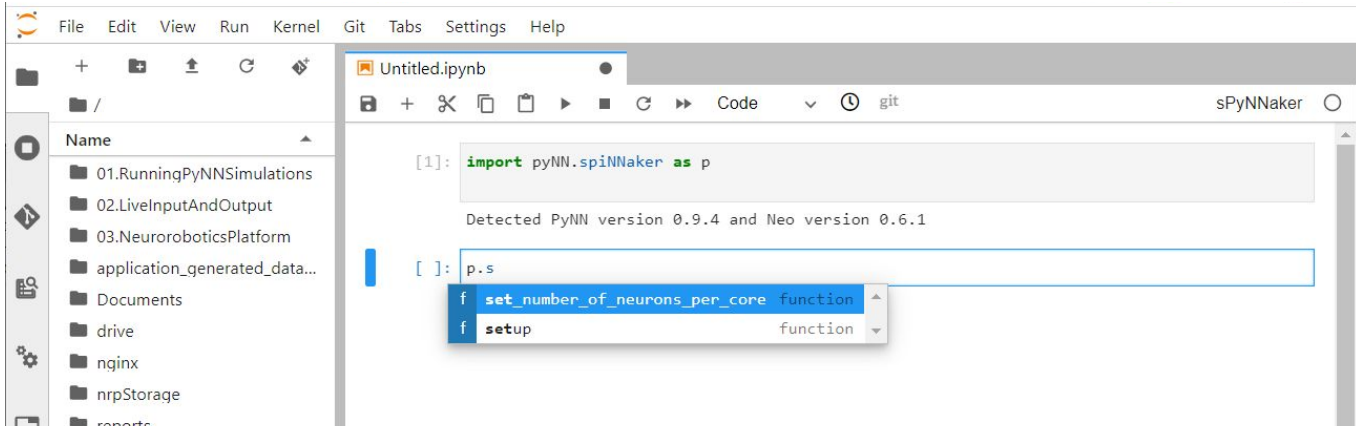
File Edit View Run Kernel Git Tabs Settings Help

New
New Launcher Ctrl+Shift+L
Open from Path...
New View for Notebook
New Console for Notebook
Close Tab Alt+W
Close and Shutdown Notebook Ctrl+Shift+Q
Close All Tabs
Save Notebook Ctrl+S
Save Notebook As... Ctrl+Shift+S
Save All
Reload Notebook from Disk
Revert Notebook to Checkpoint
Rename Notebook...
Download
Export Notebook As...
Print... Ctrl+P
Hub Control Panel
Log Out

sPyNNaker

0 s_ 1 sPyNNaker | Idle Saving completed Mode: Command Ln 1, Col 1 Untitled.ipynb

Jupyter Autocomplete



Jupyter Function Help

The screenshot shows a Jupyter Notebook window titled 'Untitled.ipynb'. The menu bar includes File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. The left sidebar shows a file explorer with a tree view containing folders like '01.RunningPyNNSimulations', '02.LiveInputAndOutput', '03.NeuroroboticsPlatform', and files like 'SpacInvaders.ipynb' and 'SynfireExample.ipynb'. The main area contains a code cell with the following content:

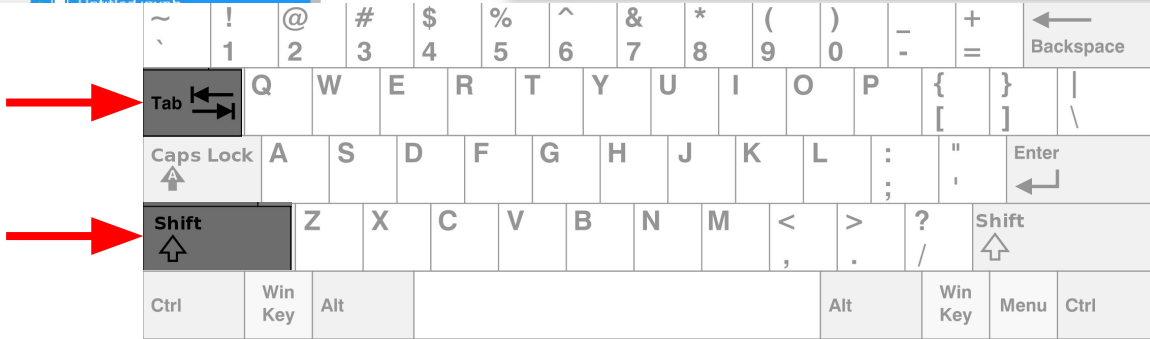
```
[1]: import pyNN.spinNaker as p
```

Below the code, it says 'Detected PyNN version 0.9.4 and Neo version 0.6.1'. A second code cell is active, containing:

```
[ ]: p.setup()
```

A help tooltip is displayed over the `p.setup()` call, showing the function signature and its parameters:

```
Signature:
p.setup(
    timestep=0.1,
    min_delay='auto',
    max_delay=10.0,
    graph_label=None,
    database_socket_addresses=None,
    extra_algorithm_xml_paths=None,
    extra_mapping_inputs=None,
    extra_mapping_algorithms=None,
    extra_pre_run_algorithms=None,
    extra_post_run_algorithms=None,
    extra_load_algorithms=None,
    time_scale_factor=None,
    n_chips_required=None,
    n_boards_required=None.
```



Install Libraries

The screenshot shows a Jupyter Notebook interface. The top menu bar includes File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. The left sidebar displays a file explorer with a list of folders and files, including '01.RunningPyNNSimulations', '02.LiveInputAndOutput', '03.NeuroboticsPlatform', 'application_generated_data...', 'Documents', 'drive', 'nginx', 'nrpStorage', 'reports', 'sPyNNaker', 'sPyNNakerGit', 'work', 'SpaceInvaders.ipynb', 'SynfireExample.ipynb', and 'Untitled.ipynb'. The main workspace shows a code cell with the command `!pip install gym`. The status bar at the bottom indicates 'Saving completed', 'Mode: Edit', 'Ln 1, Col 17', and 'Untitled.ipynb'.

Terminal Access

The image shows the JupyterLab Launcher interface. On the left is a file browser showing a directory structure with folders like '01.RunningPyNNSimulations', '02.LiveInputAndOutput', '03.NeuroroboticsPlatform', and files like 'Untitled.ipynb'. The main area is titled 'Launcher' and contains three sections: 'Notebook', 'Console', and 'Other'. Each section has four icons representing different environments or files. A tooltip 'Start a new terminal session' is visible over the 'Terminal' icon in the 'Other' section.

File Browser (Left Panel):

- Name
- 01.RunningPyNNSimulations
- 02.LiveInputAndOutput
- 03.NeuroroboticsPlatform
- application_generated_data_fi...
- Documents
- drive
- nginx
- nrpStorage
- reports
- sPyNNaker
- sPyNNakerGit
- work
- SpaceInvaders.ipynb
- SynfireExample.ipynb
- Untitled.ipynb

Launcher (Main Panel):

- Notebook**
 - Python 3
 - Python 2
 - sPyNNaker
 - sPyNNakerGit
- Console**
 - Python 3
 - Python 2
 - sPyNNaker
 - sPyNNakerGit
- Other**
 - Terminal
 - Text File
 - Markdown File
 - Show Contextual Help

Terminal tooltip: Start a new terminal session

Bottom status bar: 0 \$ _ 0

Bottom right corner: Launcher

Terminal Access

The screenshot displays the JupyterLab environment. On the left, a file browser shows a directory structure with folders like '01.RunningPyNNSimulations', '02.LiveInputAndOutput', '03.NeuroboticsPlatform', 'application_generated_data_fi...', 'Documents', 'drive', 'nginx', 'nrpStorage', 'reports', 'sPyNNaker', 'sPyNNakerGit', and 'work'. There are also Jupyter Notebook files: 'SpaceInvaders.ipynb', 'SynfireExample.ipynb', and 'Untitled.ipynb' (which is selected). The main area on the right is a terminal window titled 'spinnaker@a901776c2bd3: ~ X'. The terminal shows the prompt 'spinnaker@a901776c2bd3:~\$' with a cursor. The top menu bar includes 'File', 'Edit', 'View', 'Run', 'Kernel', 'Git', 'Tabs', 'Settings', and 'Help'. The bottom status bar shows '1 s_ 0' and the terminal path 'spinnaker@a901776c2bd3: ~'.

Start the NRP...

```

File Edit View Run Kernel Git Tabs Settings Help
+
/
Name
01.RunningPyNNSimulations
02.LiveInputAndOutput
03.NeuroboticsPlatform
application_generated_data_fi...
Documents
drive
nginx
nrpStorage
reports
sPyNNaker
sPyNNakerGit
work
SpaceInvaders.ipynb
SynfireExample.ipynb
Untitled.ipynb

roscore http://127.0.0.1:11311 X
spinnaker@a901776c2bd3:~$ cle-nginx
* Restarting nginx nginx [ OK ]
spinnaker@a901776c2bd3:~$ cle-start
[1] 1870
... logging to /home/spinnaker/.ros/log/2263c840-19d4-11eb-9668-0242ac11001c/roslaunch-a901776c2bd3-1870.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

[2] 1878
started roslaunch server http://127.0.0.1:35140/
ros_comm version 1.14.6

SUMMARY
=====

PARAMETERS
* /roscpp: melodic
* /rosversion: 1.14.6

NODES

auto-starting new master
process[master]: started with pid [1887]
ROS_MASTER_URI=http://127.0.0.1:11311/

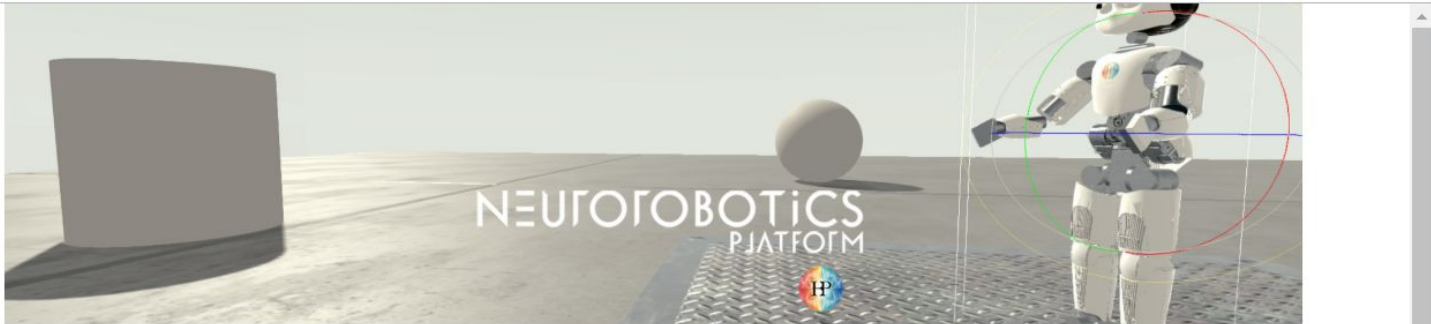
setting /run_id to 2263c840-19d4-11eb-9668-0242ac11001c
[ INFO] [1603968448.201297625]: Waiting For connections on 0.0.0.0:8081
process[rosout-1]: started with pid [1904]
started core service [/rosout]
[3] 1919
[4] 1956
[uWSGI] getting INI configuration from /home/spinnaker/.local/etc/nginx/uwsgi-nrp.ini
*** Starting uWSGI 2.0.15-debian (64bit) on [Thu Oct 29 10:47:29 2020] ***
compiled with version: 7.3.0 on 28 September 2018 15:41:15

1 $ 0
roscore http://127.0.0.1:11311/

```

Connect to the NRP

<https://spinn-20.cs.man.ac.uk/user/<username>/proxy/9000/#/esv-private>



Local authentication form

LOGIN

NRP Storage

The screenshot shows a terminal window with the following output:

```

spinnaker@a901776c2bd3:~$ cle-nginx
* Restarting nginx nginx
spinnaker@a901776c2bd3:~$ cle-start
[1] 1870
... logging to /home/spinnaker/.ros/log/2263c840-19d4-11eb-9668-0242ac11001c/roslaunch-a901776c2bd3-1870.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

[2] 1878
started roslaunch server http://127.0.0.1:35140/
ros_comm version 1.14.6

SUMMARY
=====

PARAMETERS
* /roscpp: melodic
* /rosversion: 1.14.6

NODES

auto-starting new master
process[master]: started with pid [1887]
ROS_MASTER_URI=http://127.0.0.1:11311/

setting /run_id to 2263c840-19d4-11eb-9668-0242ac11001c
[ INFO ] [1603968448.201297625]: Waiting For connections on 0.0.0.0:8081
process[rosout-1]: started with pid [1904]
started core service [/rosout]
[3] 1919
[4] 1956
[uWSGI] getting INI configuration from /home/spinnaker/.local/etc/nginx/uwsgi-nrp.ini
*** Starting uWSGI 2.0.15-debian (64bit) on [Thu Oct 29 10:47:29 2020] ***
compiled with version: 7.3.0 on 28 September 2018 15:41:15
  
```

The file explorer on the left shows the directory structure, with a red arrow pointing to the `nrpStorage` folder.

Logout!

The image shows a terminal window with a context menu open. The menu items are:

- New
- New Launcher (Ctrl+Shift+L)
- Open from Path...
- New View for
- New Console for Activity
- Close Tab (Alt+W)
- Close and Shutdown Terminal (Ctrl+Shift+Q)
- Close All Tabs
- Save (Ctrl+S)
- Save As... (Ctrl+Shift+S)
- Save All
- Reload from Disk
- Revert to Checkpoint
- Rename ...
- Download
- Export Notebook As...
- Print... (Ctrl+P)
- Hub Control Panel
- Log Out

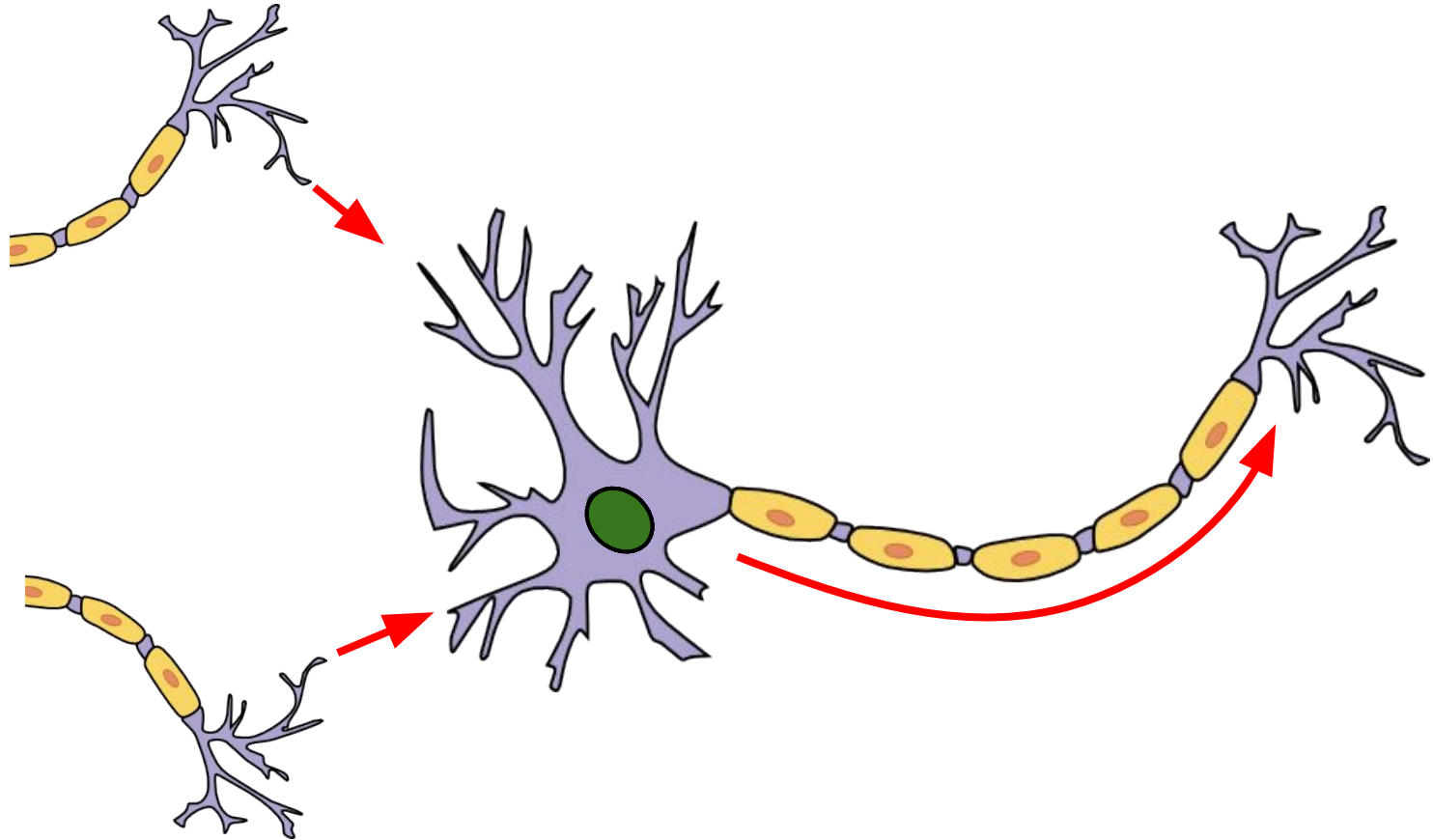
The terminal output in the background includes:

```

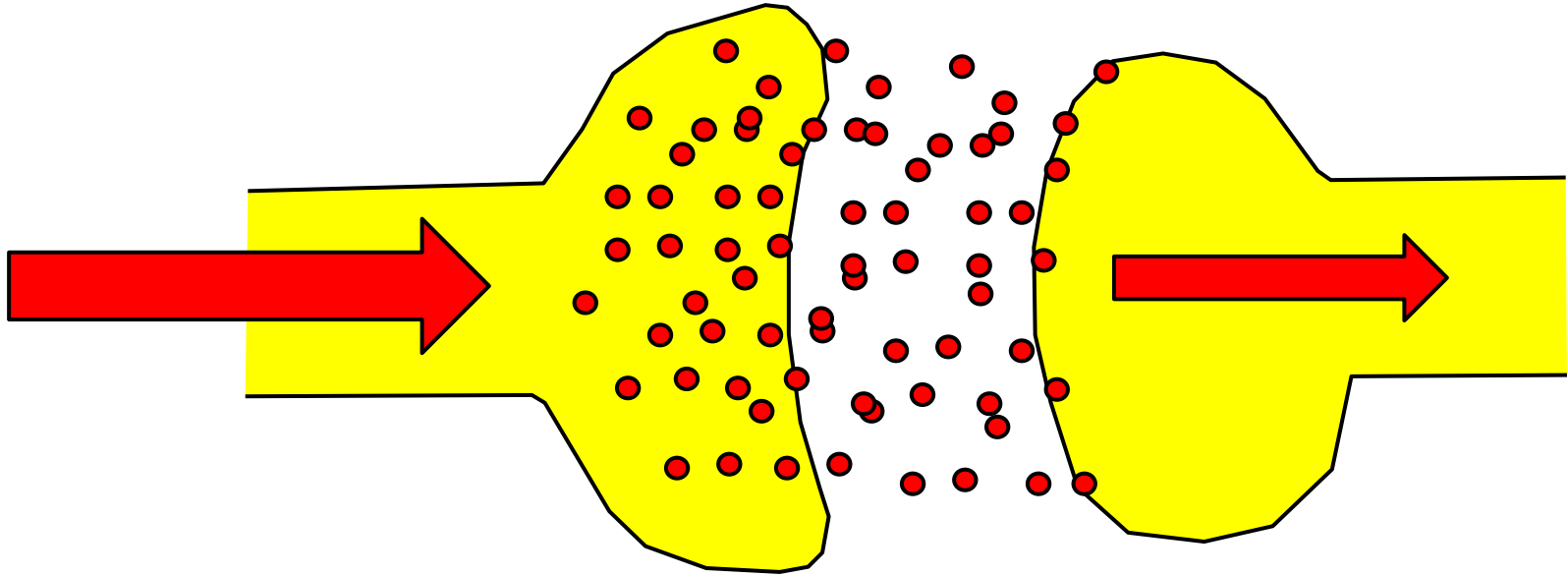
http://127.0.0.1:11311 X
1776c2bd3:~$ c le-nginx
; nginx nginx [ OK ]
1776c2bd3:~$ c le-start
o /home/spinnaker/.ros/log/2263c840-19d4-11eb-9668-0242ac11001c/roslaunch-a901776c2bd3-1870.log
directory for disk usage. This may take a while.
to interrupt
; log file disk usage. Usage is <1GB.
unch server http://127.0.0.1:35140/
ion 1.14.6
o: melodic
on: 1.14.6
; new master
er]: started with pid [1887]
I=http://127.0.0.1:11311/
id to 2263c840-19d4-11eb-9668-0242ac11001c
968448.201297625]: Waiting For connections on 0.0.0.0:8081
rt-1]: started with pid [1904]
service [/rosout]
[3] 1919
[4] 1956
[uWSGI] getting INI configuration from /home/spinnaker/.local/etc/nginx/uwsgi-nrp.ini
*** Starting uWSGI 2.0.15-debian (64bit) on [Thu Oct 29 10:47:29 2020] ***
compiled with version: 7.3.0 on 28 September 2018 15:41:15
roscore http://127.0.0.1:11311/
  
```

Spiking Neural Networks

Spiking Neural Networks

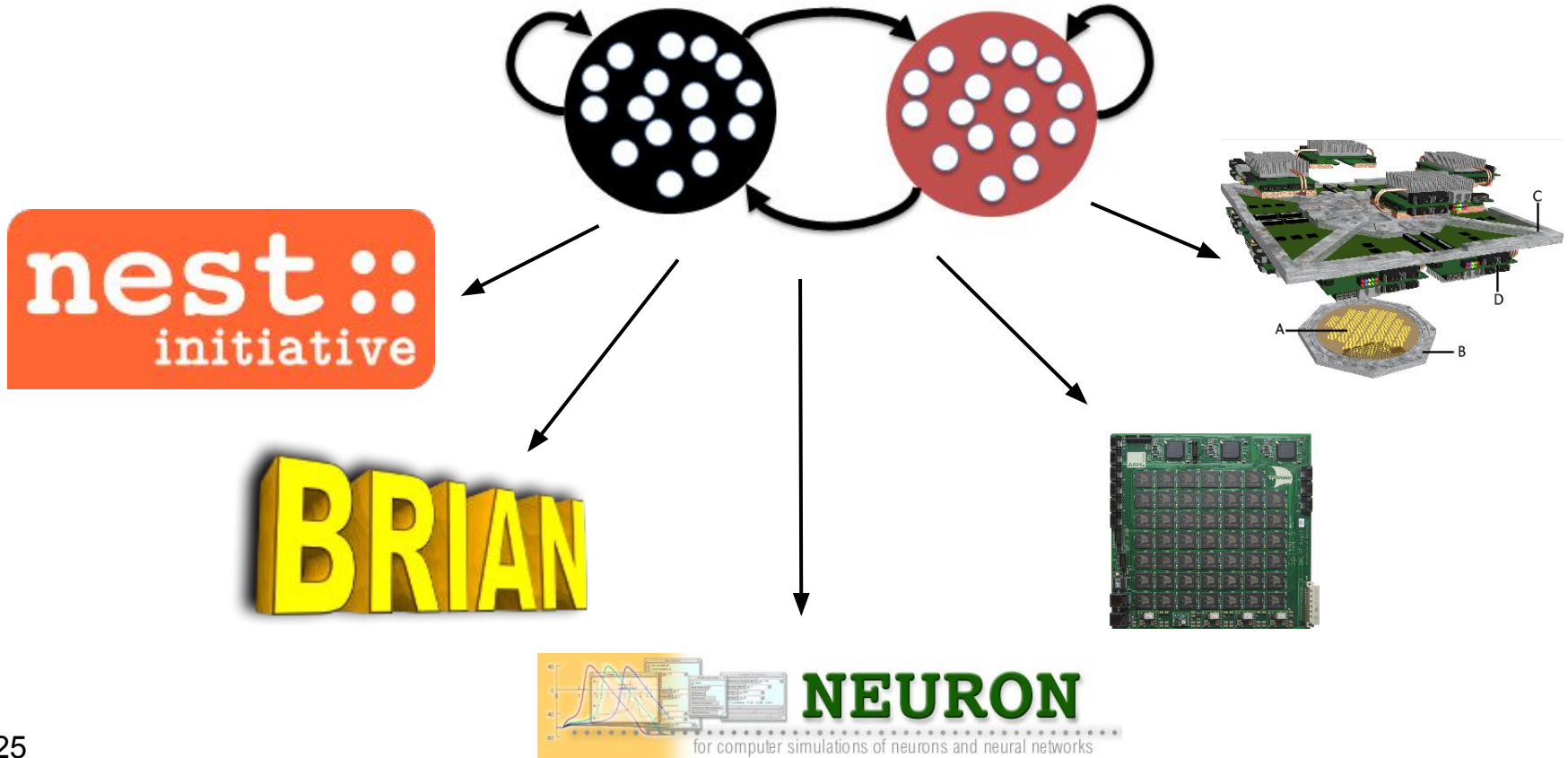


Spiking Neural Networks



PyNN

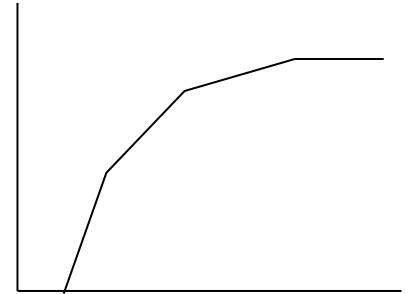
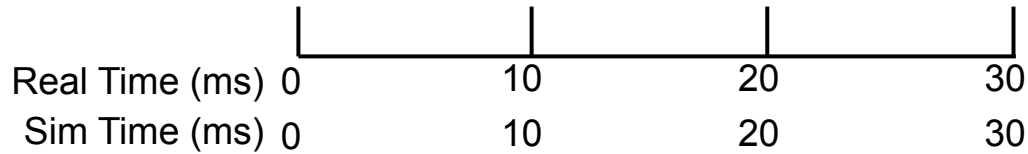
What is PyNN?



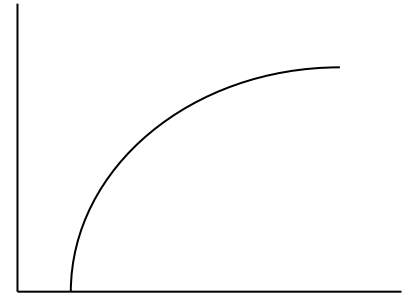
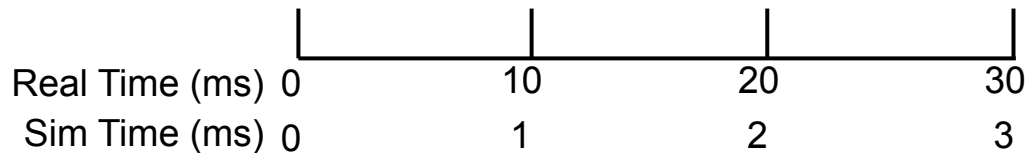
PyNN - Setup

```
import pyNN.spiNNaker as p
```

```
p.setup(timestep=1.0)
```

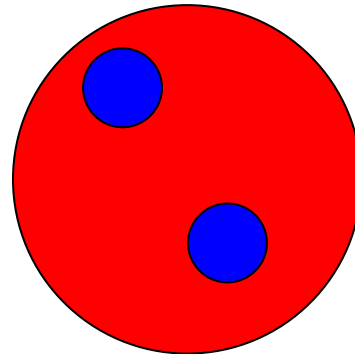
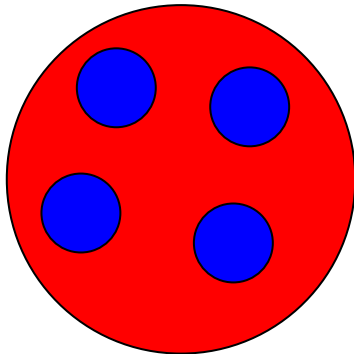


```
p.setup(timestep=0.1)
```



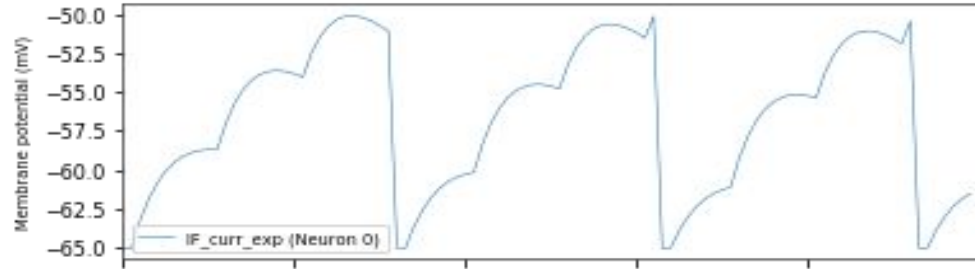
PyNN - Populations

```
pop_1 = p.Population(  
    4, p.IF_curr_exp(), label="Fred")  
pop_2 = p.Population(  
    2, p.IF_curr_exp(), label="Bob")
```

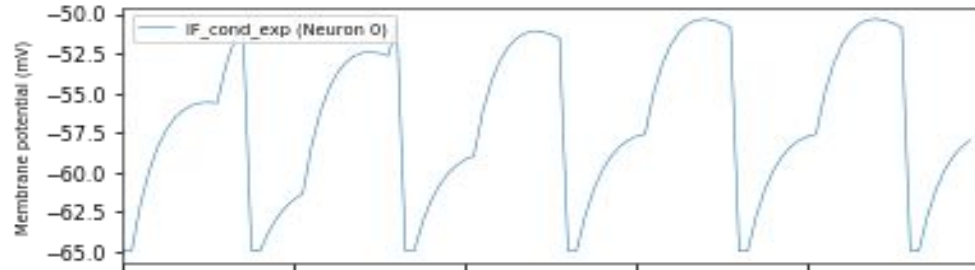


PyNN Populations - Models

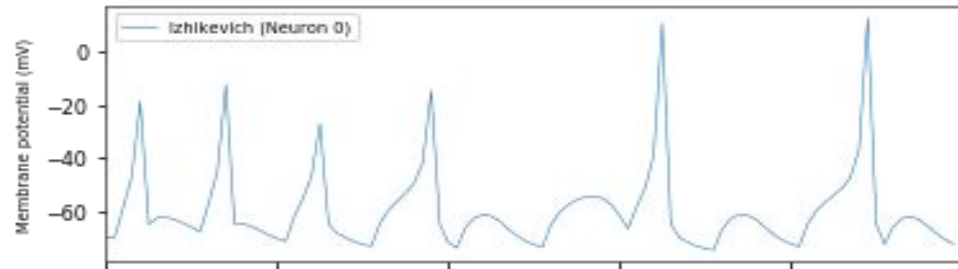
`p.IF_curr_exp(...)`



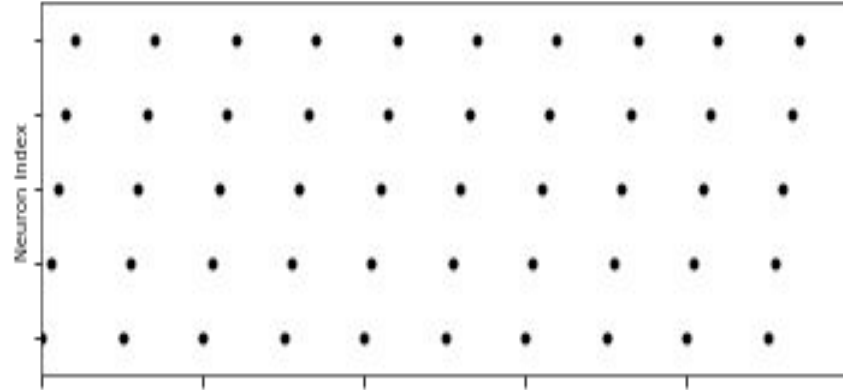
`p.IF_cond_exp(...)`



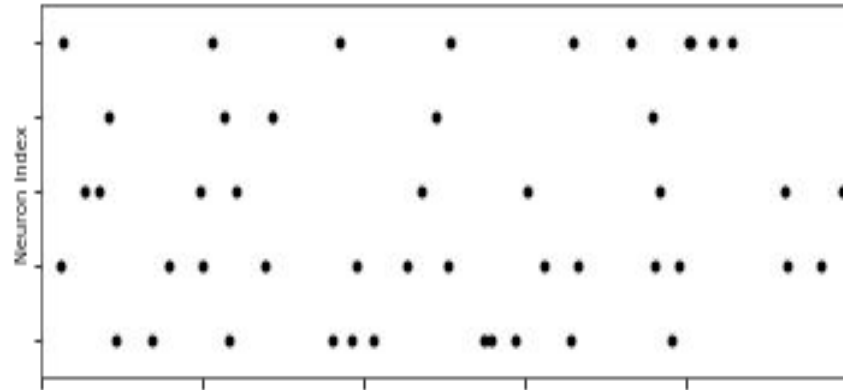
`p.Izhikevich(...)`



```
p.SpikeSourceArray(  
    spike_times=[...])
```

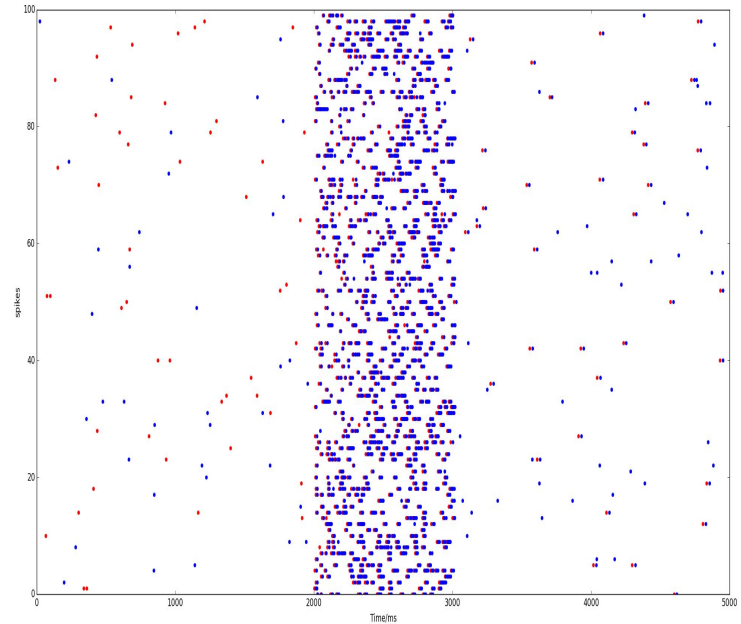
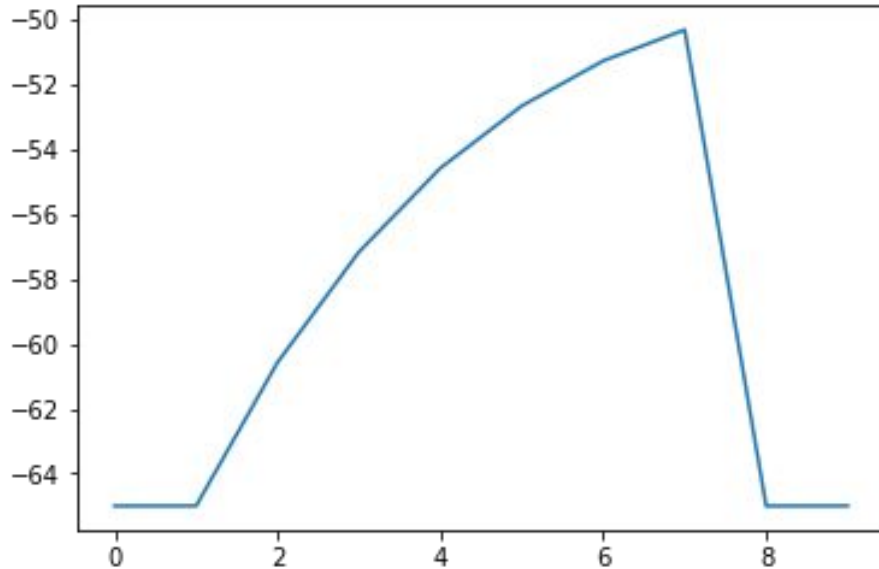


```
p.SpikeSourcePoisson(  
    rate=...)
```



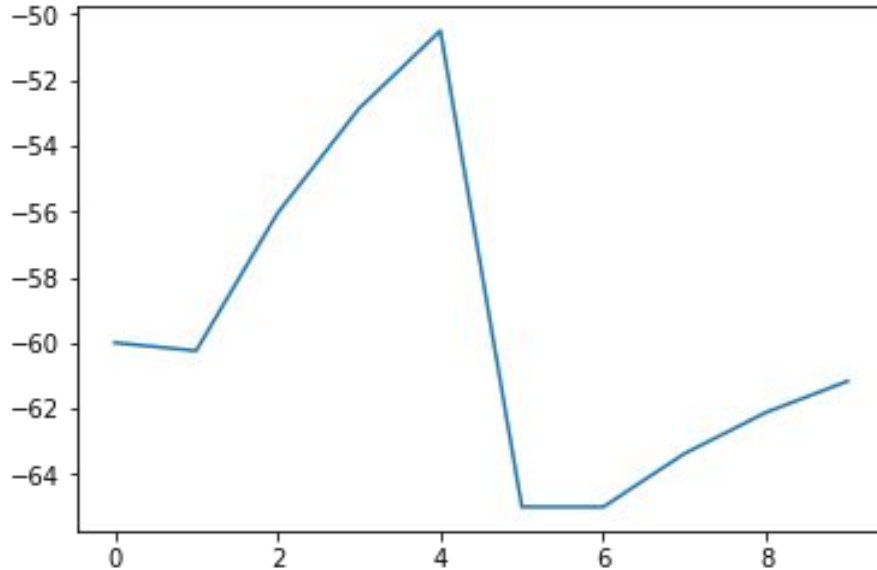
PyNN Populations - Recording

```
pop_1.record(["v", "spikes"])
```



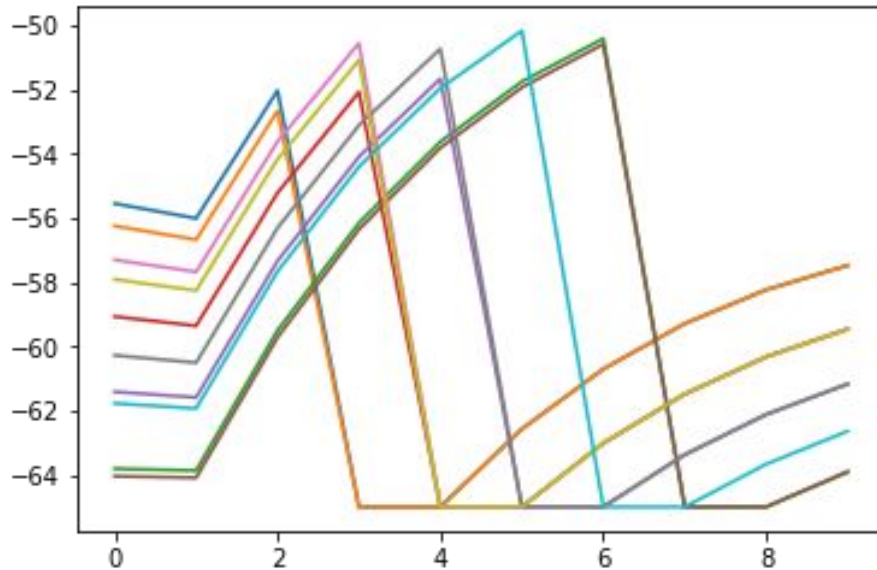
PyNN Populations - Initialize

```
pop_1.initialize(v=-60)
```



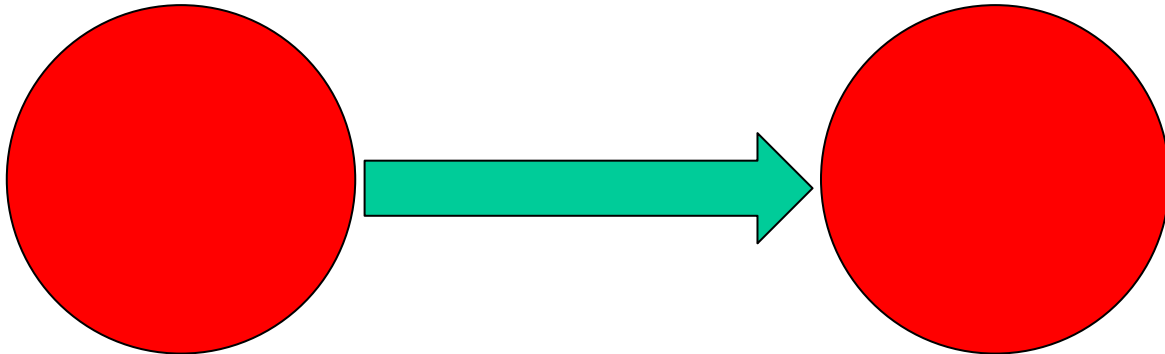
PyNN - Random Parameters

```
pop_1.initialize(v=p.RandomDistribution(  
    "uniform", low=-65.0, high=-55.0))
```



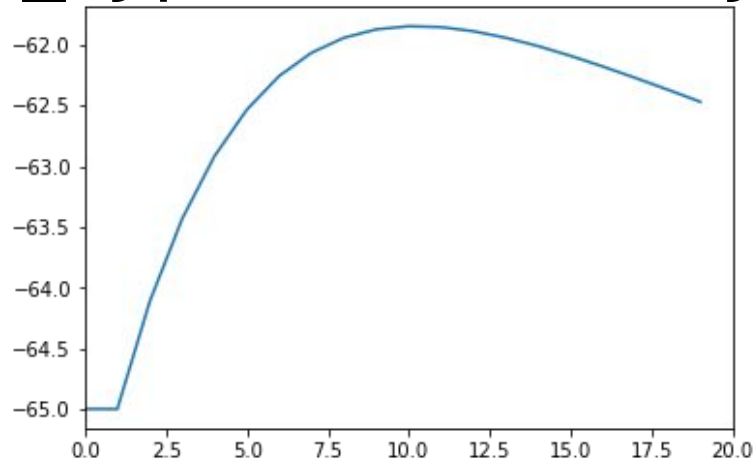
PyNN - Projections

```
proj = p.Projection(  
    pop_1, pop_2,  
    p.OneToOneConnector(),  
    p.StaticSynapse(  
        weight=1.0, delay=2.0))
```



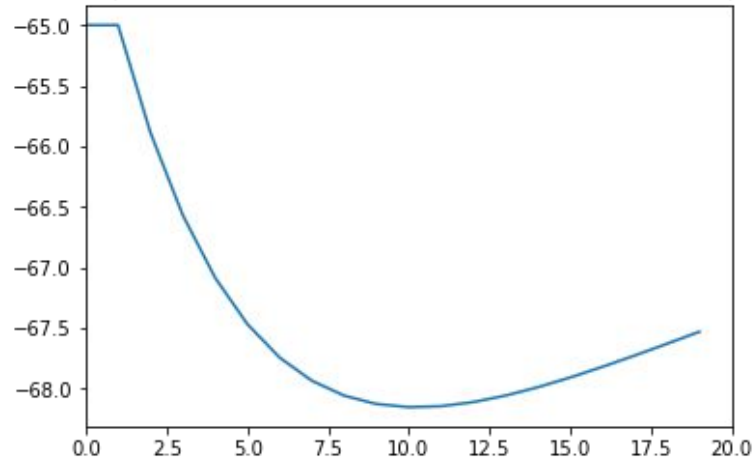
PyNN - Projections

```
proj = p.Projection(  
    pop_1, pop_2,  
    p.OneToOneConnector(),  
    p.StaticSynapse(weight=1.0),  
    receptor_type="excitatory")
```

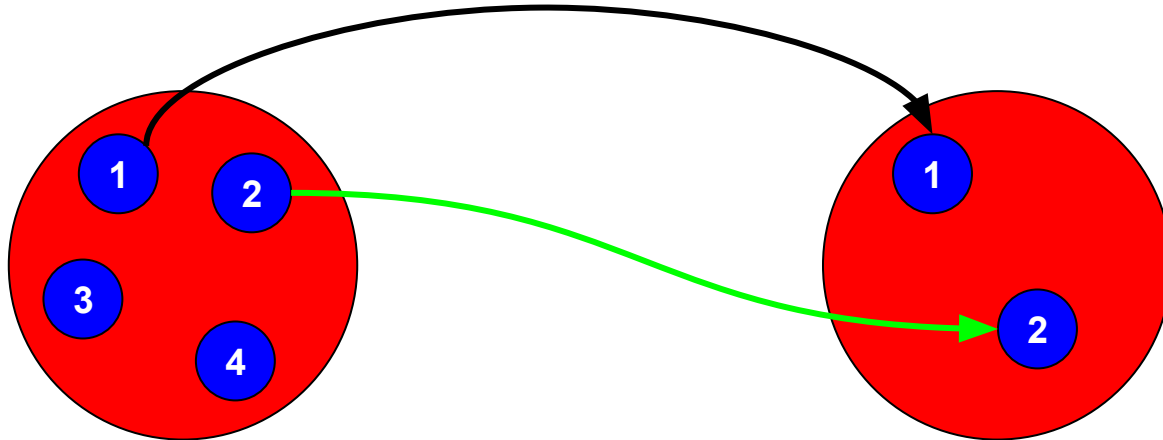


PyNN - Projections

```
proj = p.Projection(  
    pop_1, pop_2,  
    p.OneToOneConnector(),  
    p.StaticSynapse(weight=1.0),  
    receptor_type="inhibitory")
```

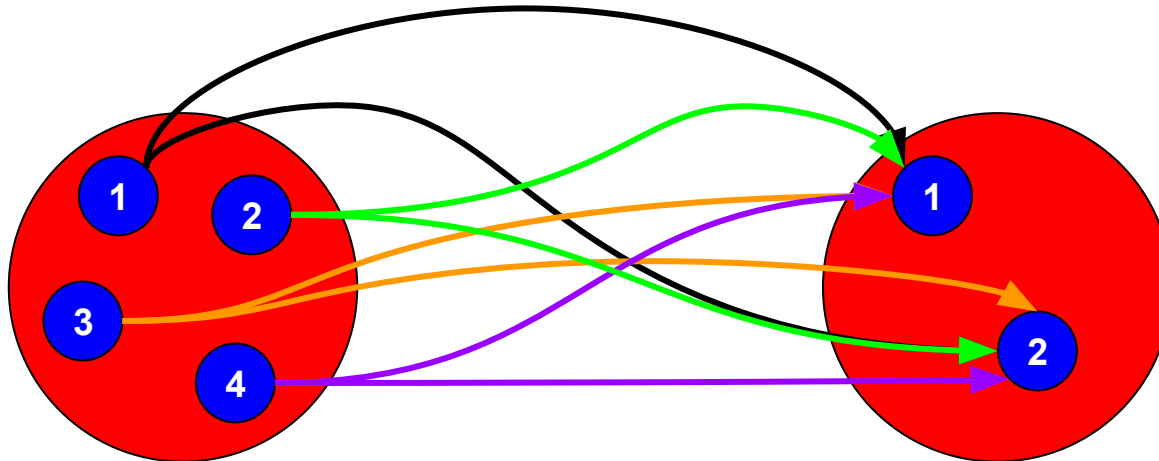


```
p.OneToOneConnector()
```



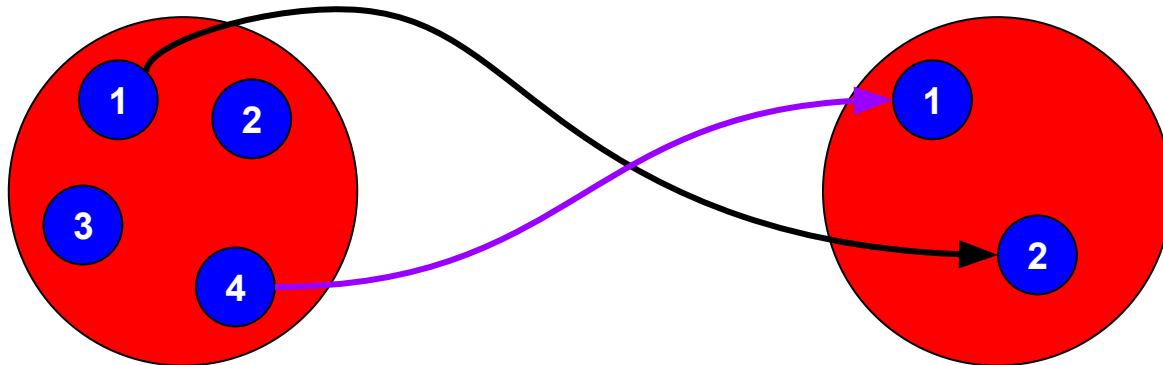
PyNN Projections - Connectors

```
p.AllToAllConnector()
```



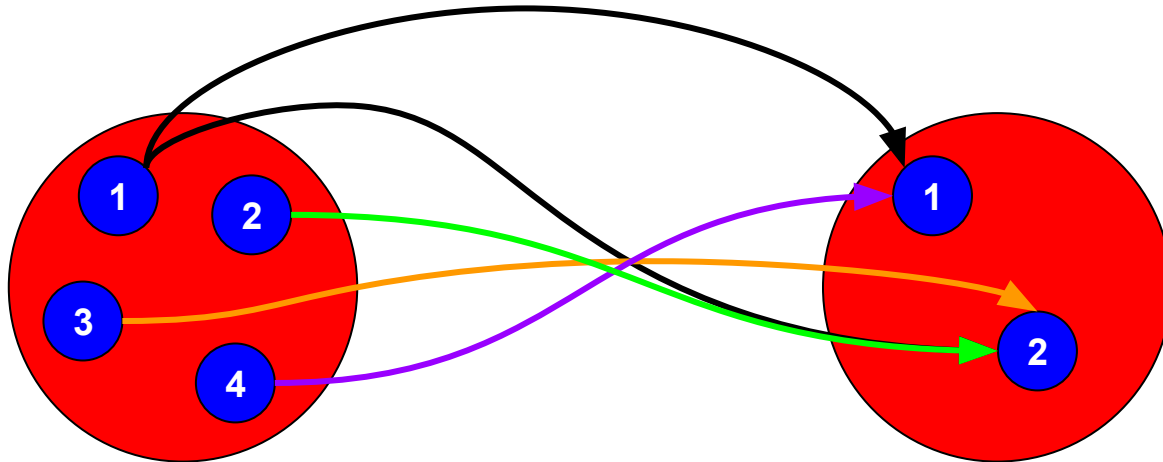
PyNN Projections - Connectors

`p.FixedProbabilityConnector(p=0.1)`



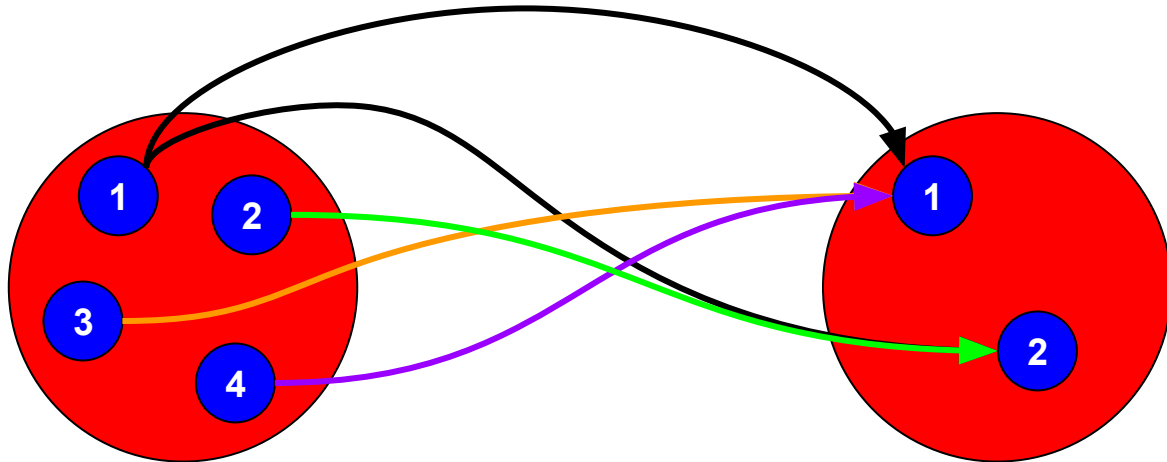
PyNN Projections - Connectors

`p.FixedProbabilityConnector(p=0.5)`



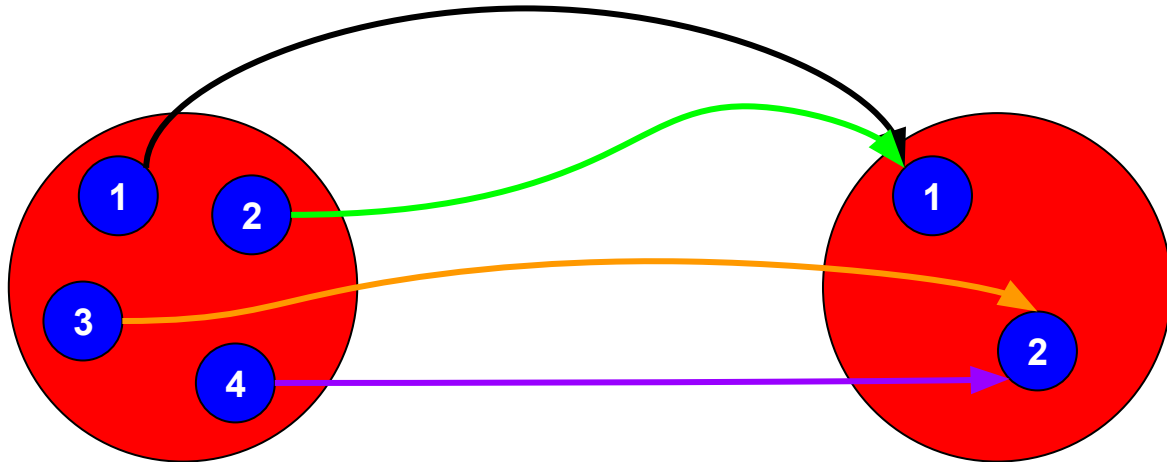
PyNN Projections - Connectors

`p.FixedTotalNumberConnector(5)`



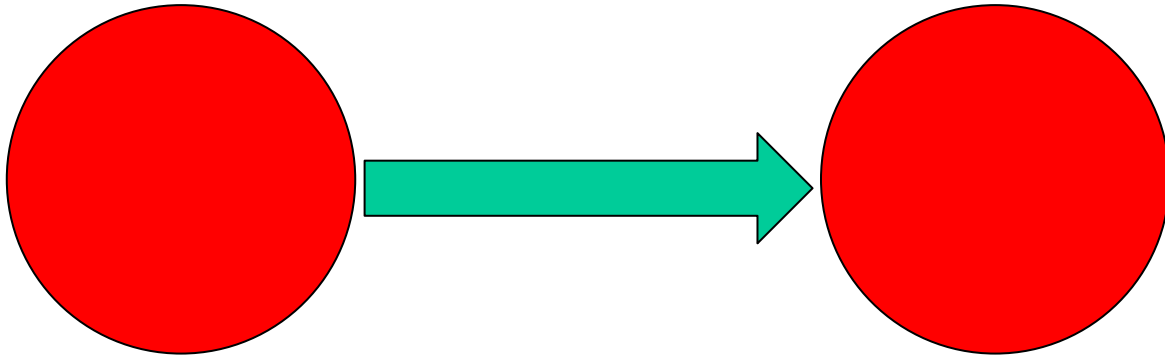
PyNN Projections - Connectors

```
p.FromListConnector(  
    [(1, 1), (2, 1), (3, 2), (4, 2)])
```



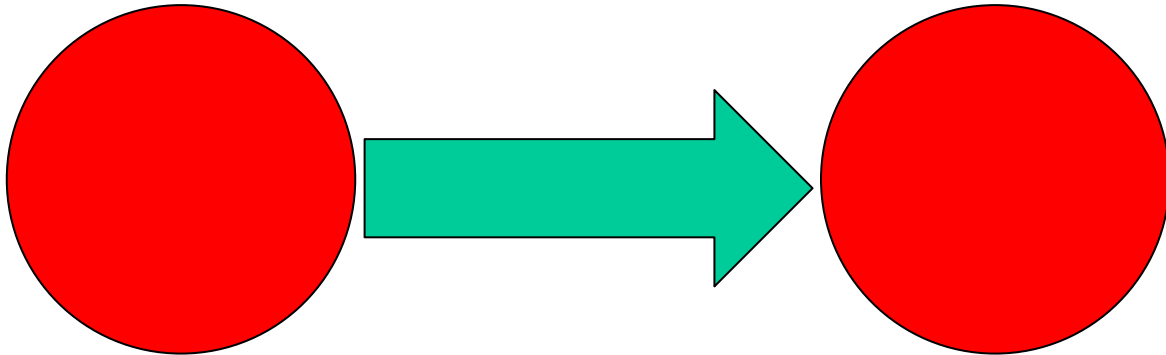
PyNN - Static Synapse Types

```
p.StaticSynapse(weight=1.0, delay=2.0)
```



PyNN - Static Synapse Types

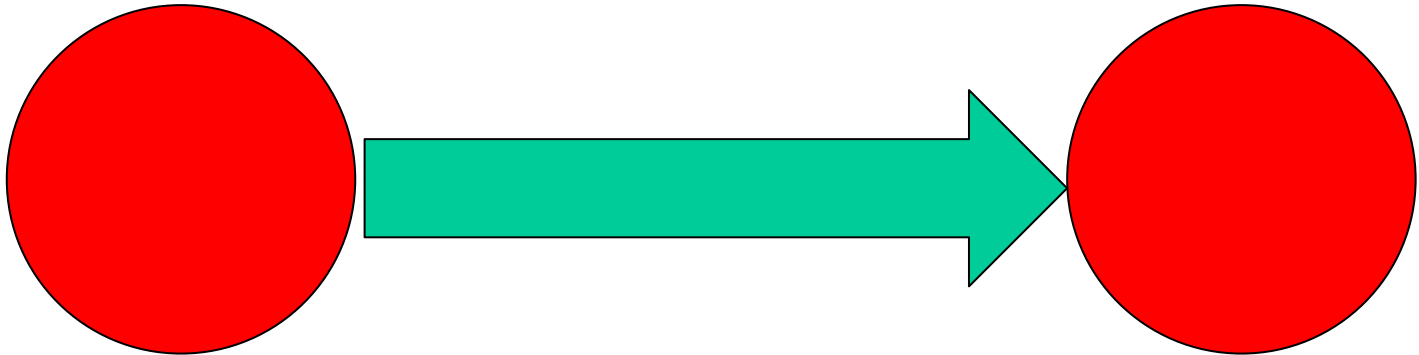
```
p.StaticSynapse(weight=5.0, delay=2.0)
```



PyNN - Static Synapse Types

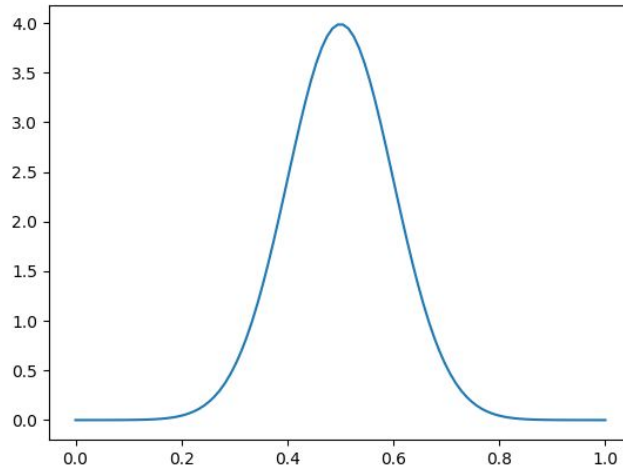
```
p.StaticSynapse(weight=5.0, delay=3.0)
```

(timestep \leq delay \leq 144 * timestep)



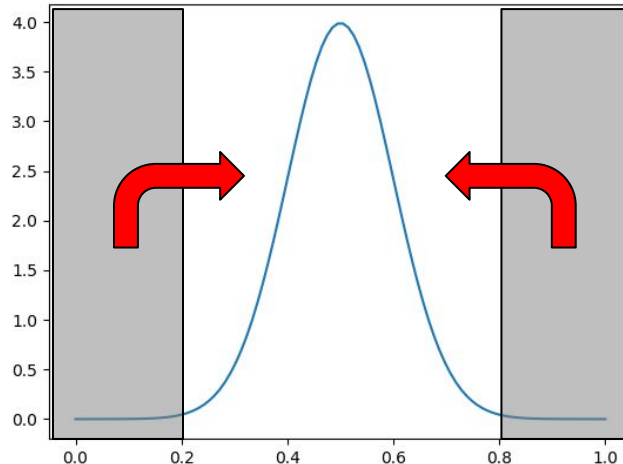
PyNN - Random Weights and Delays

```
weight_dist = p.RandomDistribution(  
    "normal", mu=0.5, sigma=0.1)  
p.StaticSynapse(  
    weight=weight_dist, delay=3.0)
```



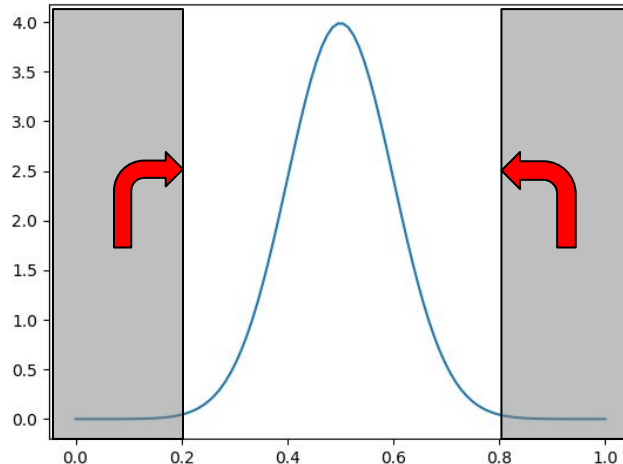
PyNN - Random Weights and Delays

```
weight_dist = p.RandomDistribution(  
    "normal_clipped",  
    mu=0.5, sigma=0.1, low=0.2 high=0.8)
```

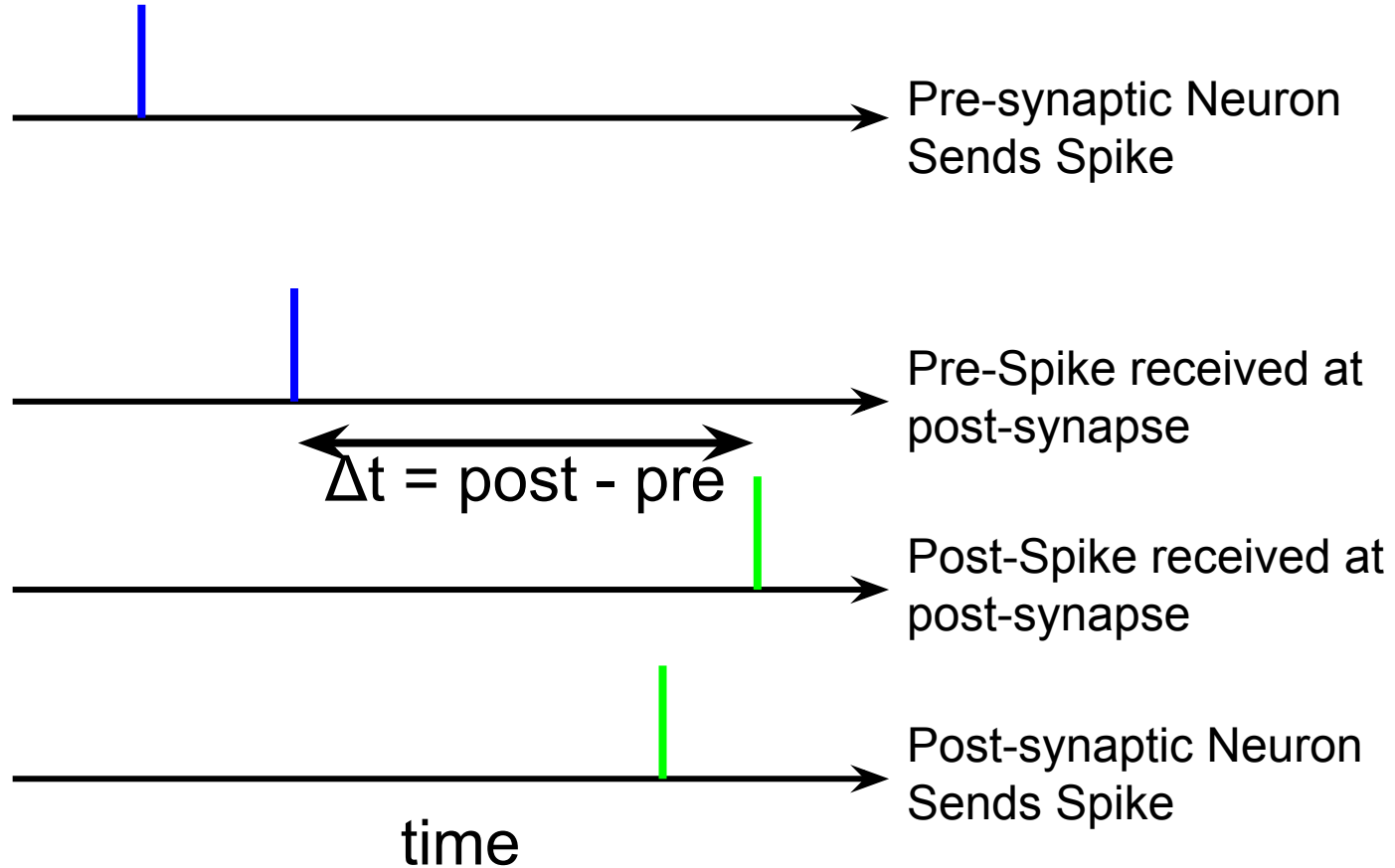
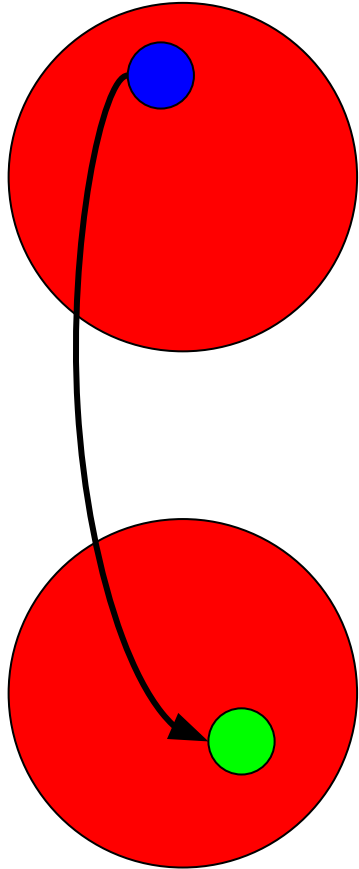


PyNN - Random Weights and Delays

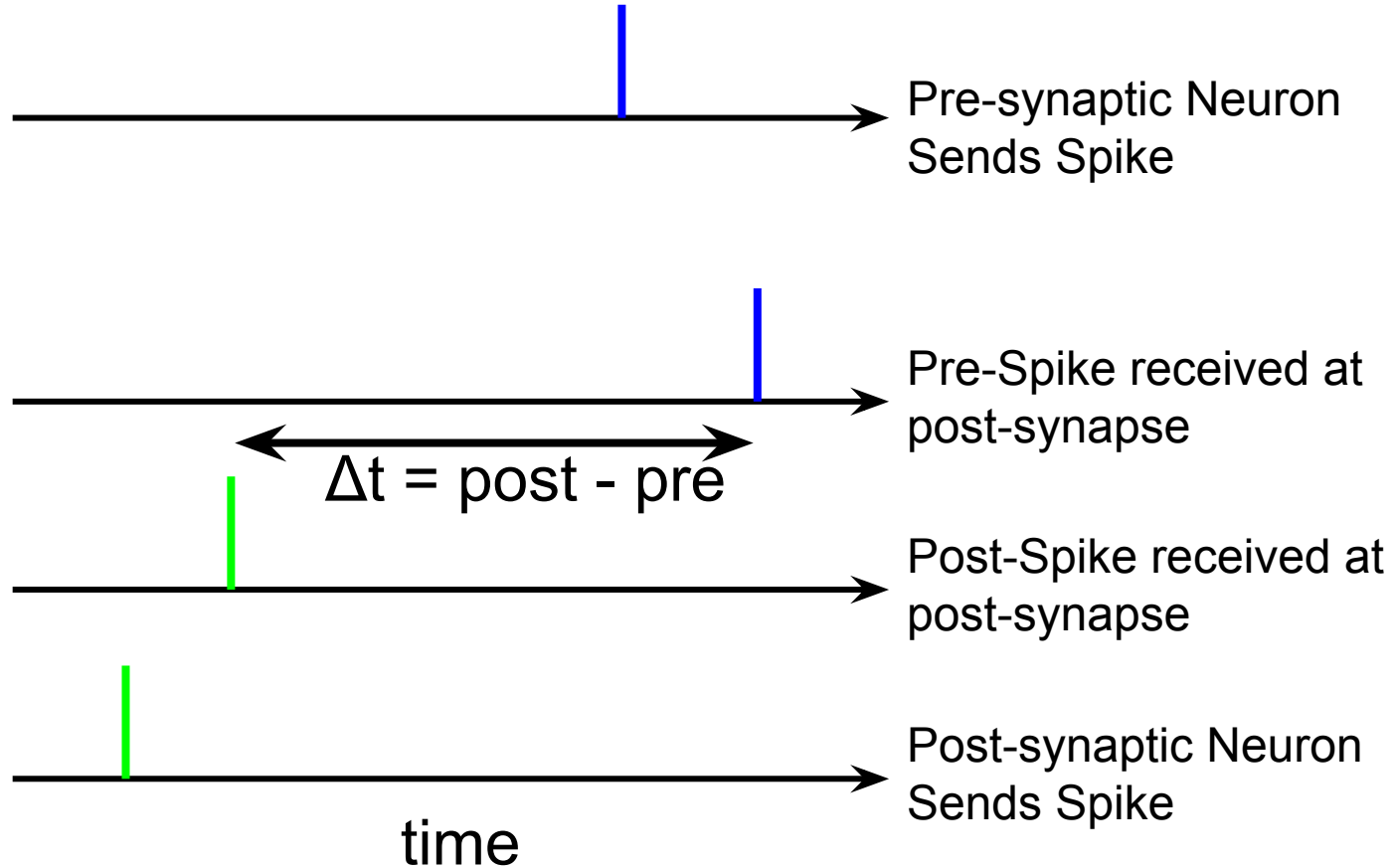
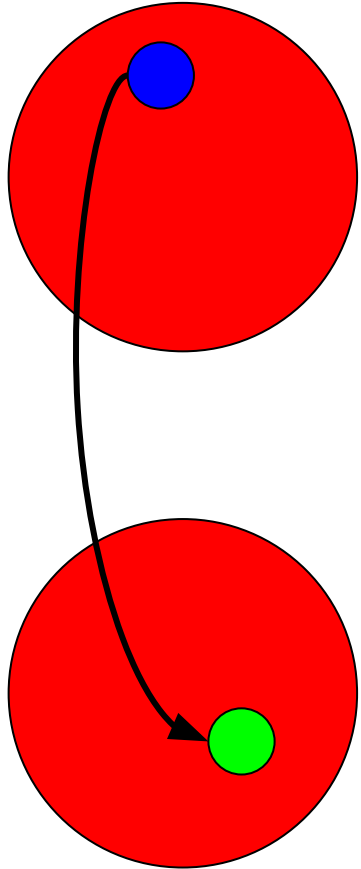
```
weight_dist = p.RandomDistribution(  
    "normal_clipped_to_boundary",  
    mu=0.5, sigma=0.1, low=0.2 high=0.8)
```



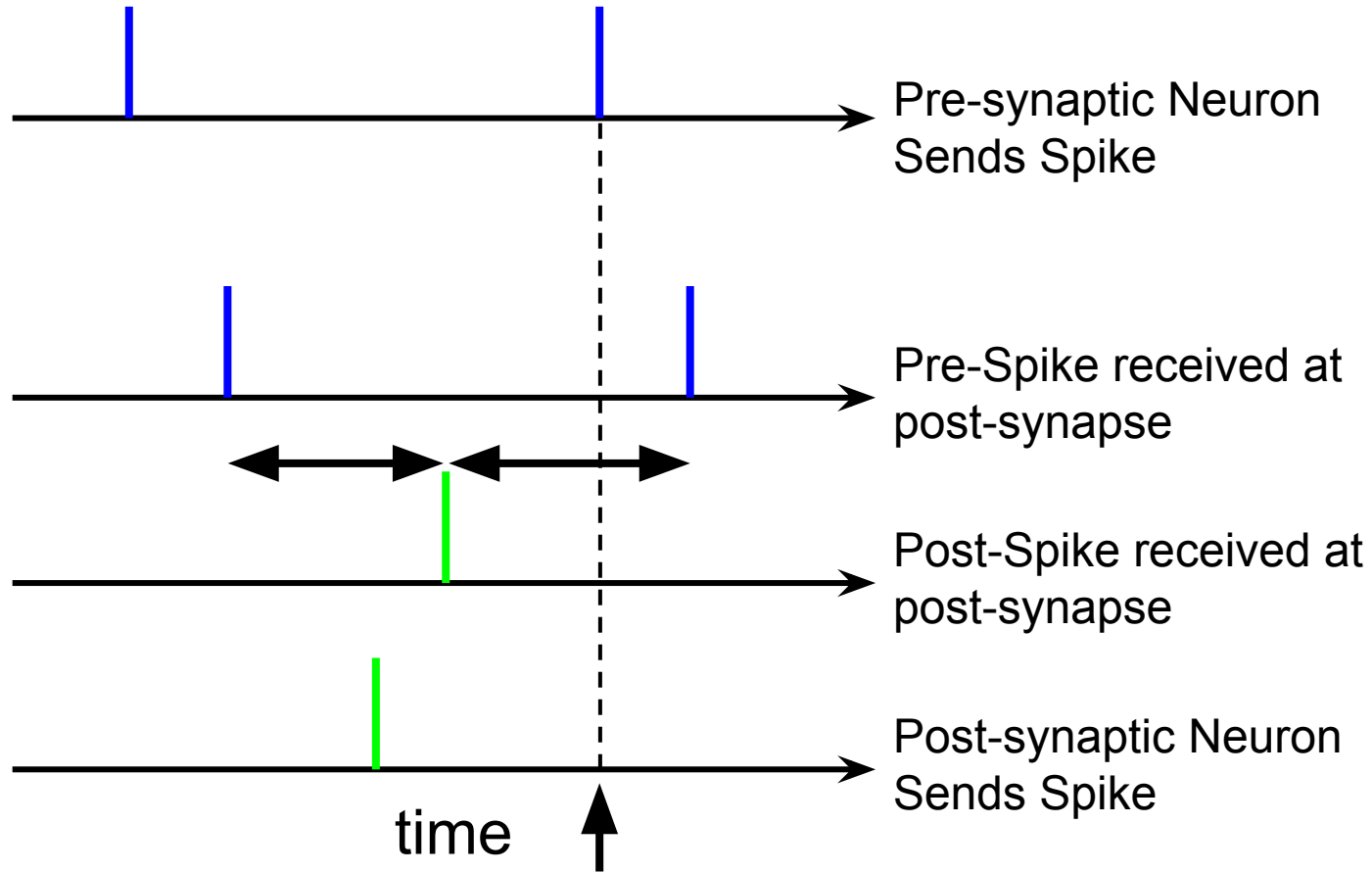
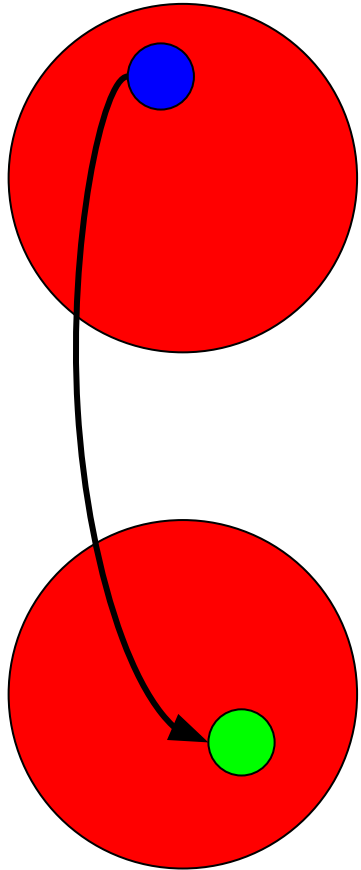
Spike Timing Dependent Plasticity



Spike Timing Dependent Plasticity

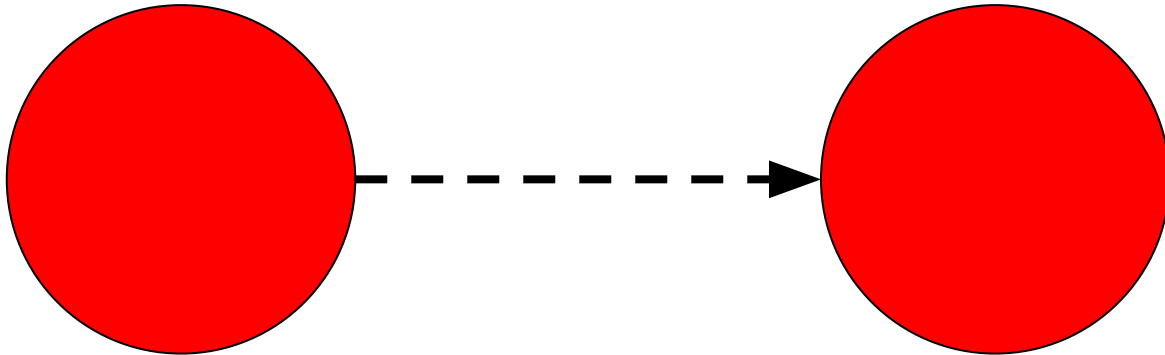


STDP - Deferred Execution



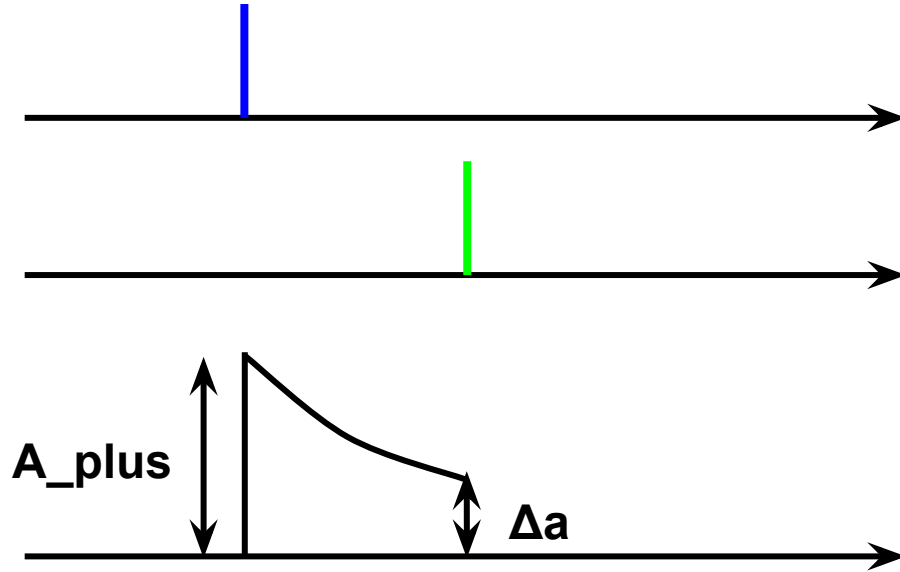
PyNN - STDP Synapse Types

```
p.STDPMechanism(  
    timing_dependence=?,  
    weight_dependence=?,  
    weight=0.0, delay=2.0)
```



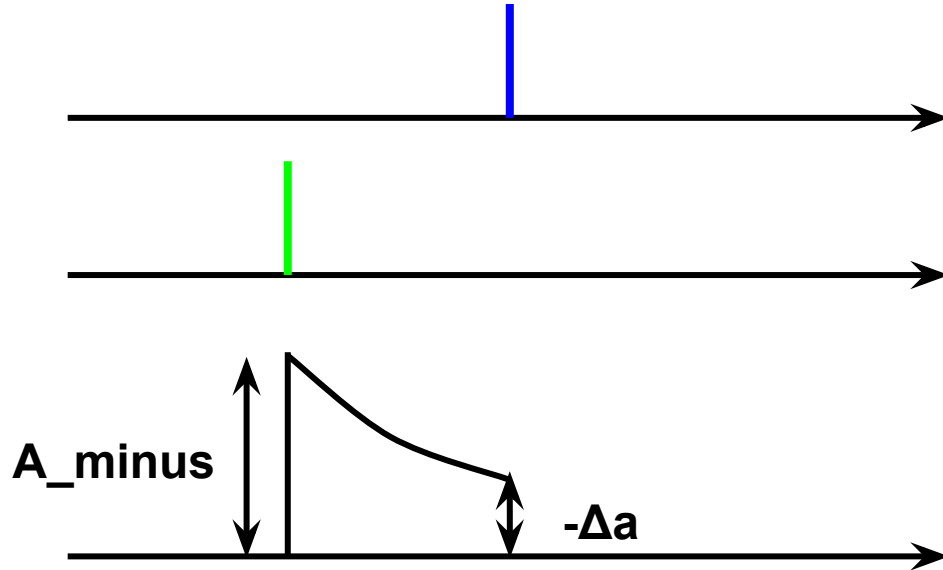
PyNN - Timing Dependence

```
sim.SpikePairRule(tau_plus=20.0, tau_minus=20.0,  
                  A_plus=0.5, A_minus=0.5)
```



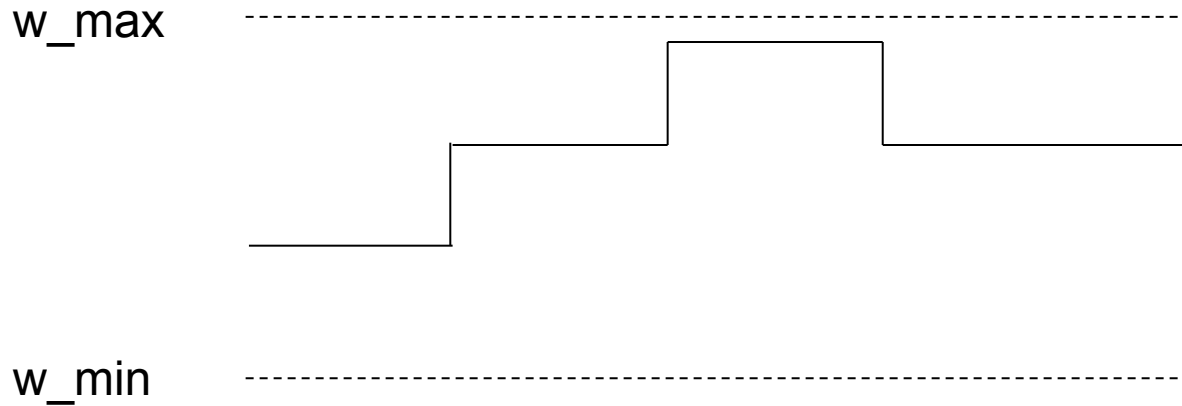
PyNN - Timing Dependence

```
sim.SpikePairRule(tau_plus=20.0, tau_minus=20.0,  
                  A_plus=0.5, A_minus=0.5)
```



PyNN - Weight Dependence

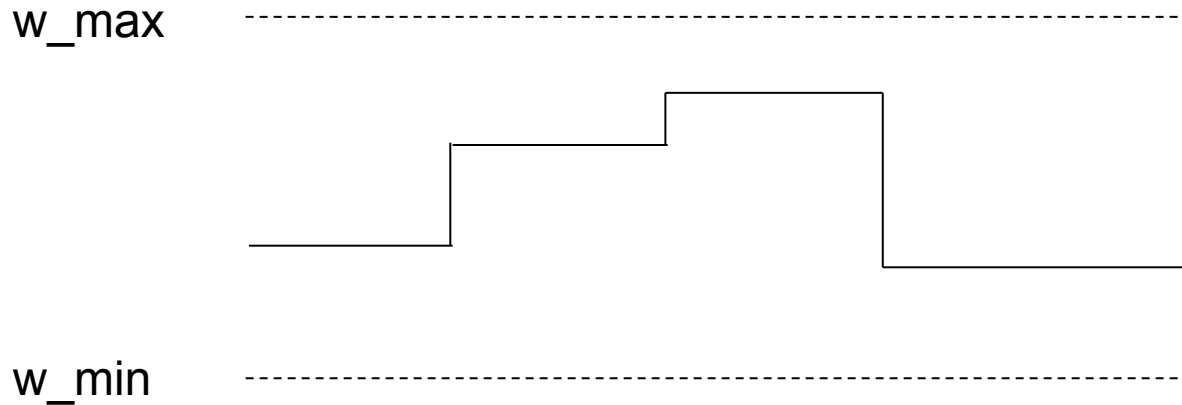
```
sim.AdditiveWeightDependence(w_max=5.0, w_min=0.0)
```



$$\Delta w = \Delta a (w_{\max} - w_{\min})$$

PyNN - Weight Dependence

```
sim.MultiplicativeWeightDependence(w_max=5.0,w_min=0.0)
```



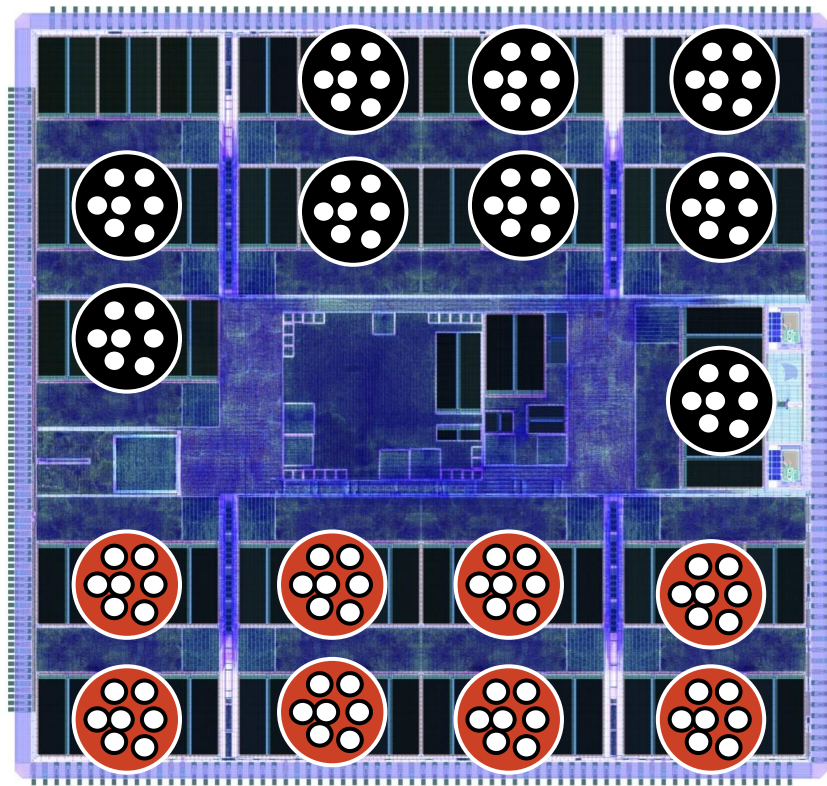
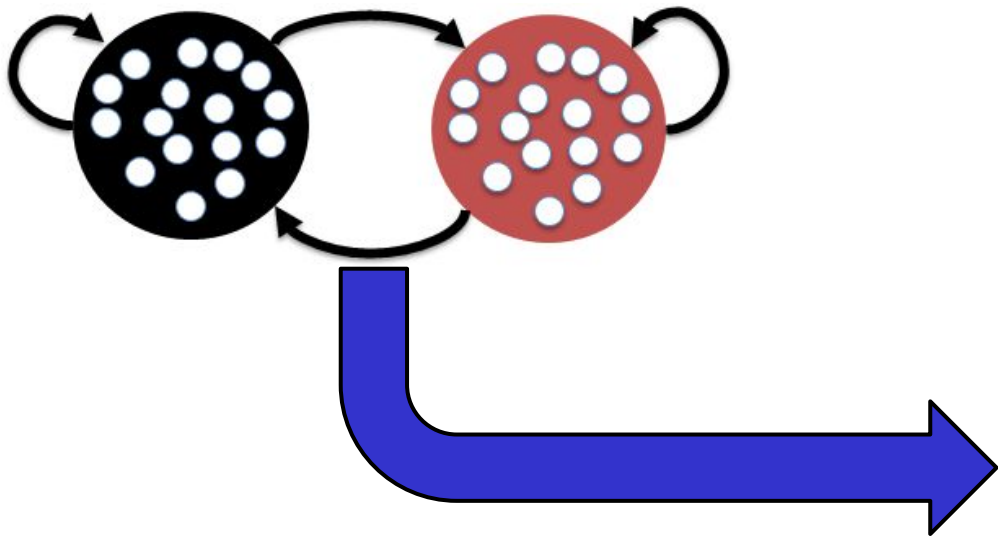
$$\Delta w = \Delta a (w - w_{\min}) \text{ if } \Delta a < 0 \text{ (Depression)}$$

$$\Delta w = \Delta a (w_{\max} - w) \text{ if } \Delta a > 0 \text{ (Potentiation)}$$

Running on SpiNNaker

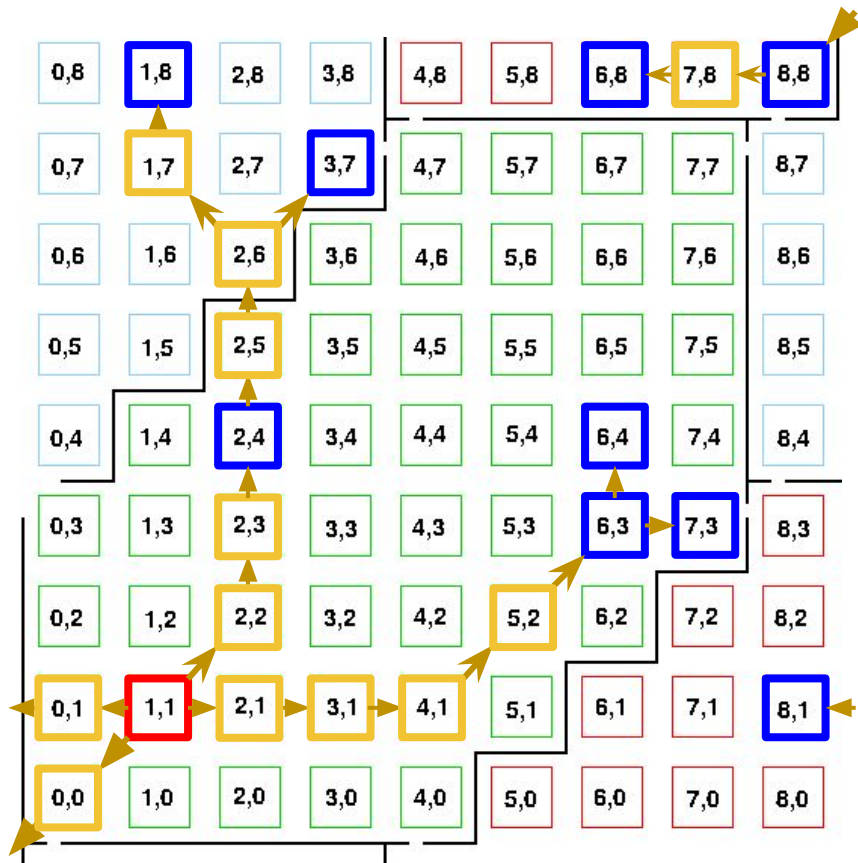
PyNN - Run

```
p.run(100)
```



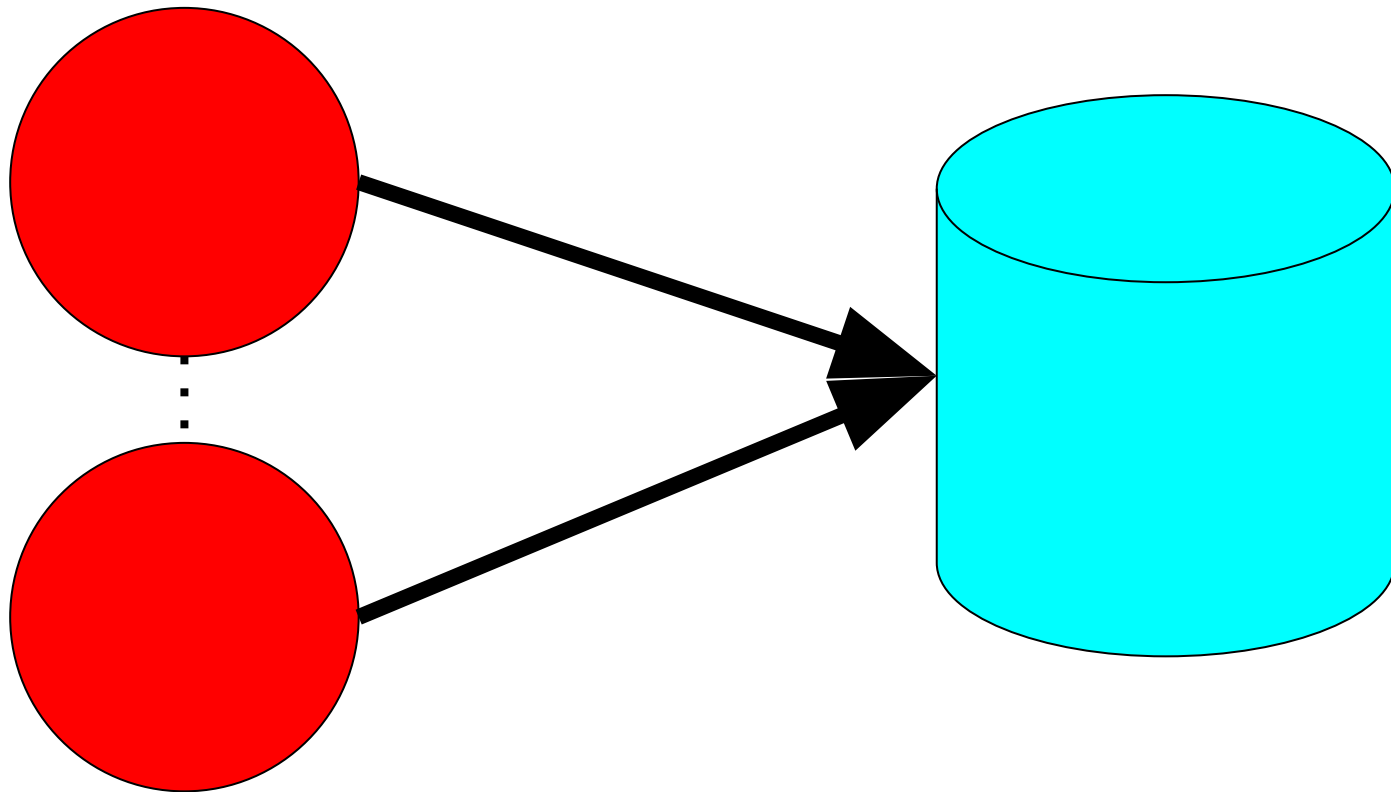
PyNN - Run

p.run(100)



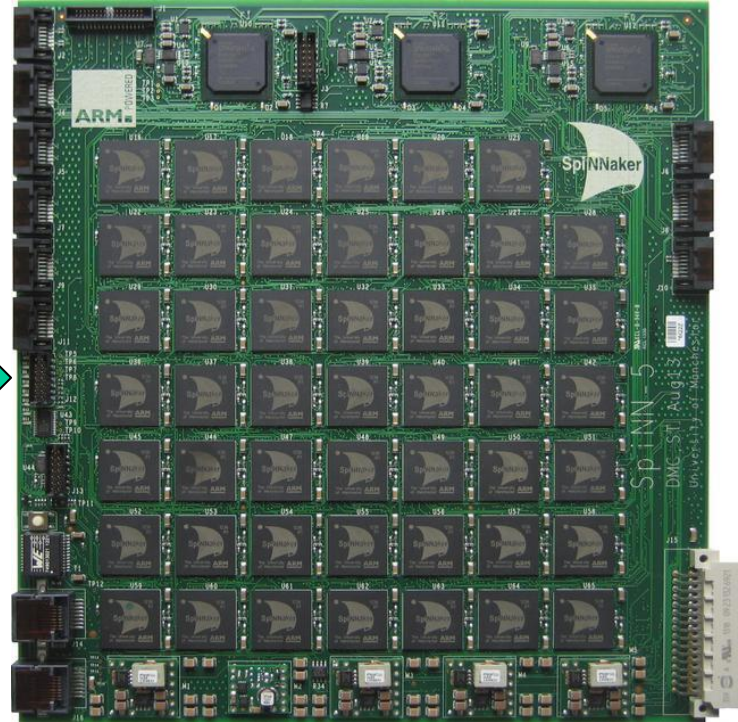
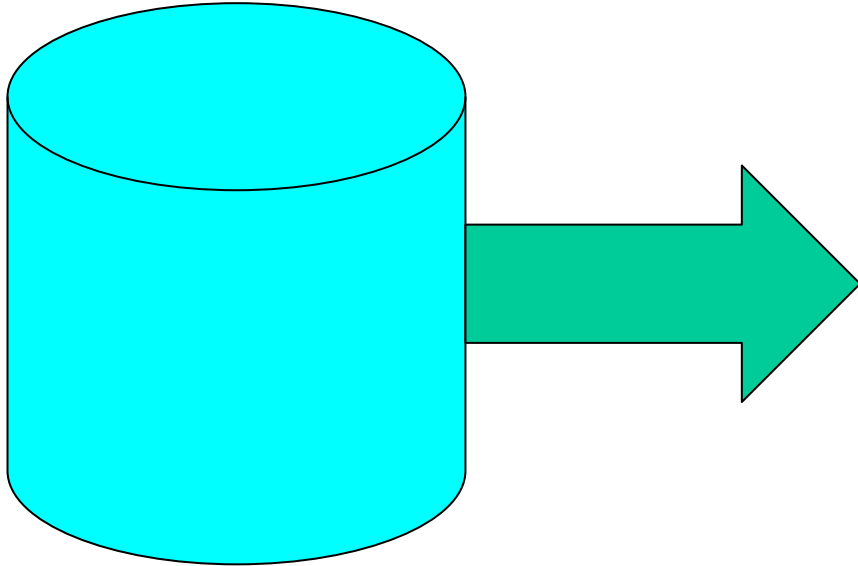
PyNN - Run

```
p.run(100)
```



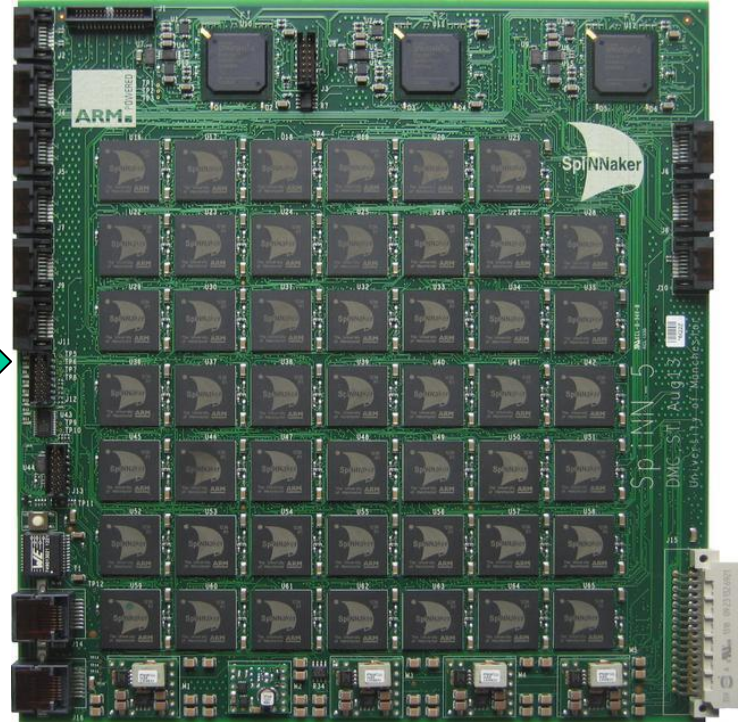
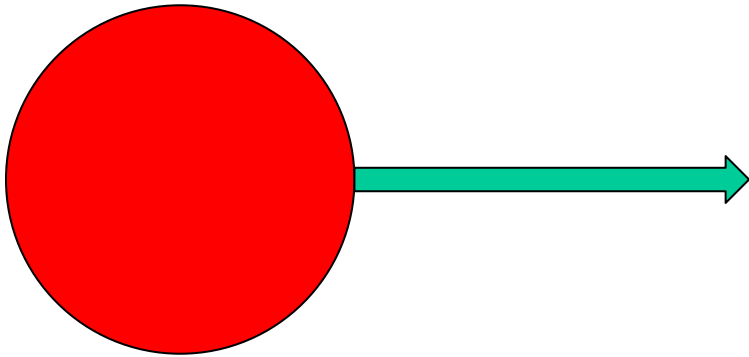
PyNN - Run

```
p.run(100)
```



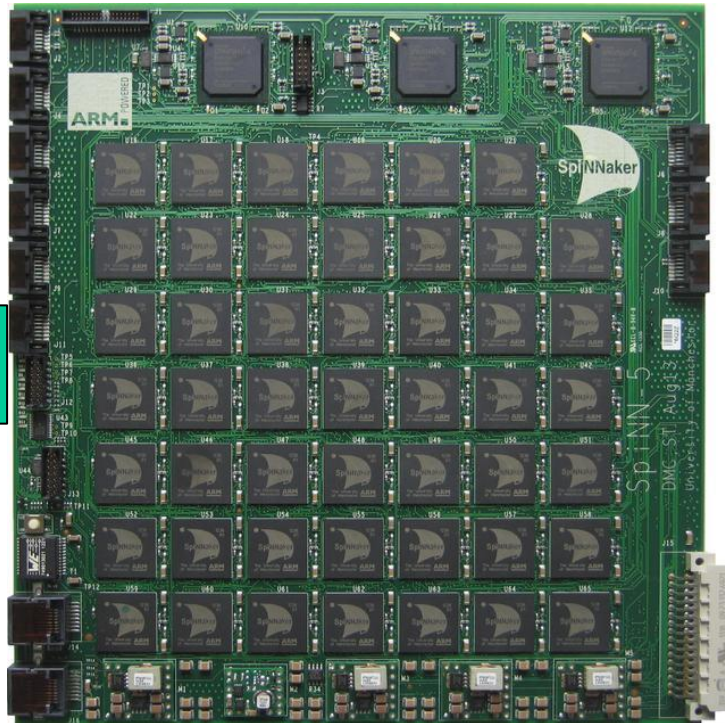
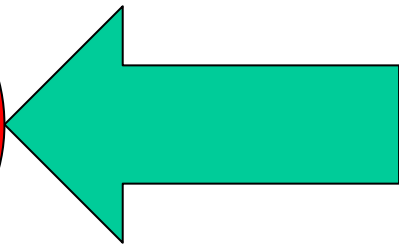
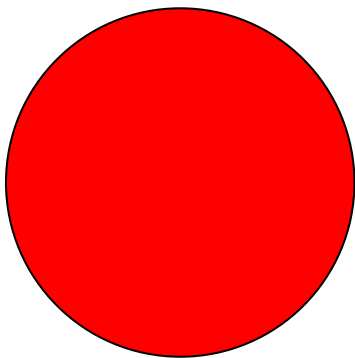
PyNN - Change and Run Again

```
pop_1.set(i_offset=5.0)  
p.run(50)
```



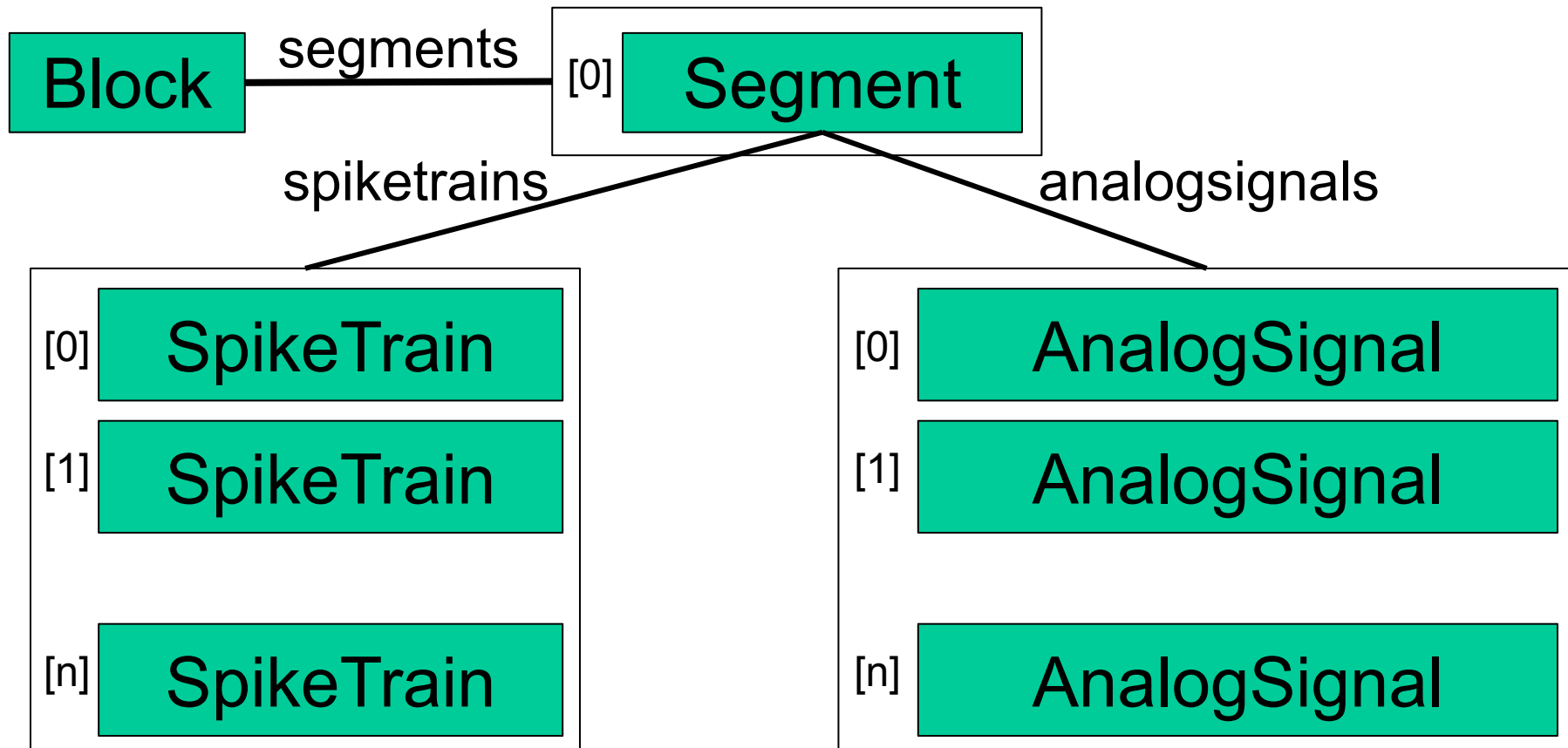
PyNN - Get Data

```
data = pop_1.get_data(["v", "spikes"])
```

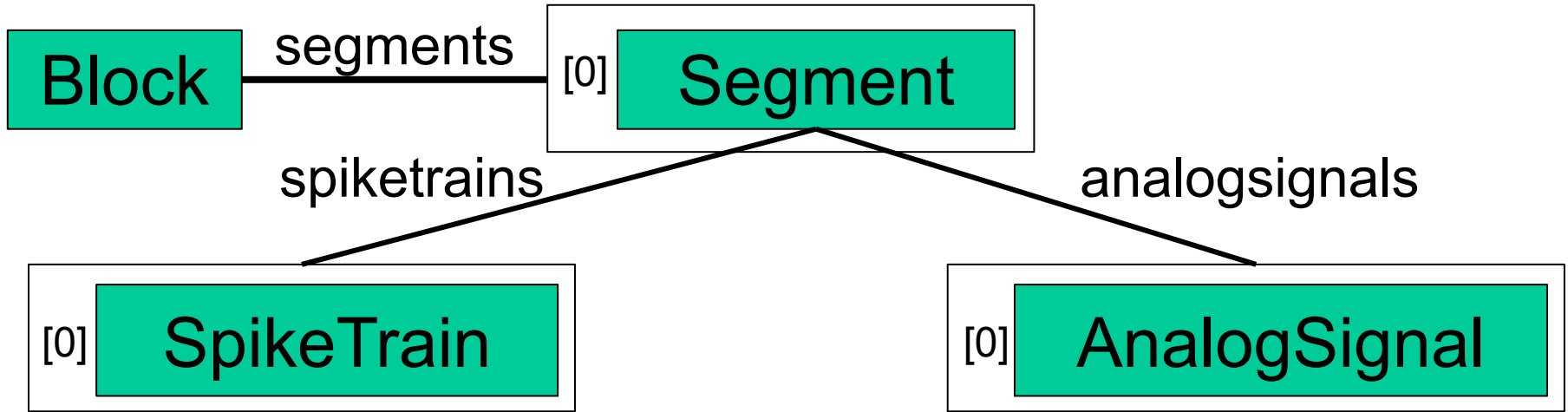


Reading Results

PyNN - Neo Data



PyNN - Neo Data



```

data = pop_1.get_data(["v", "spikes"])
v = data.segments[0].analogsignals
spikes = data.segments[0].spiketrains
  
```

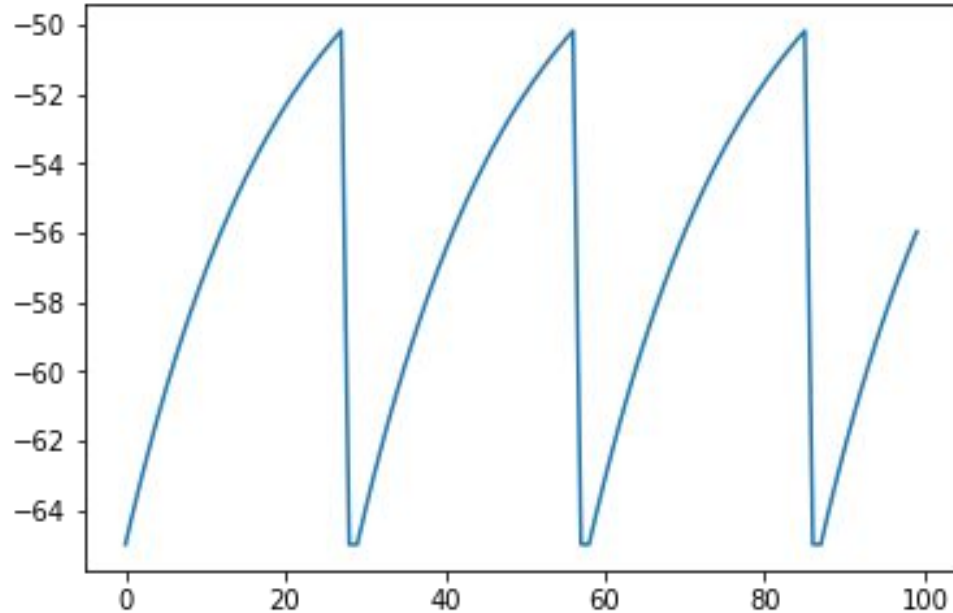
PyNN - Plotting

```
import matplotlib.pyplot as plt
```

```
plt.figure()
```

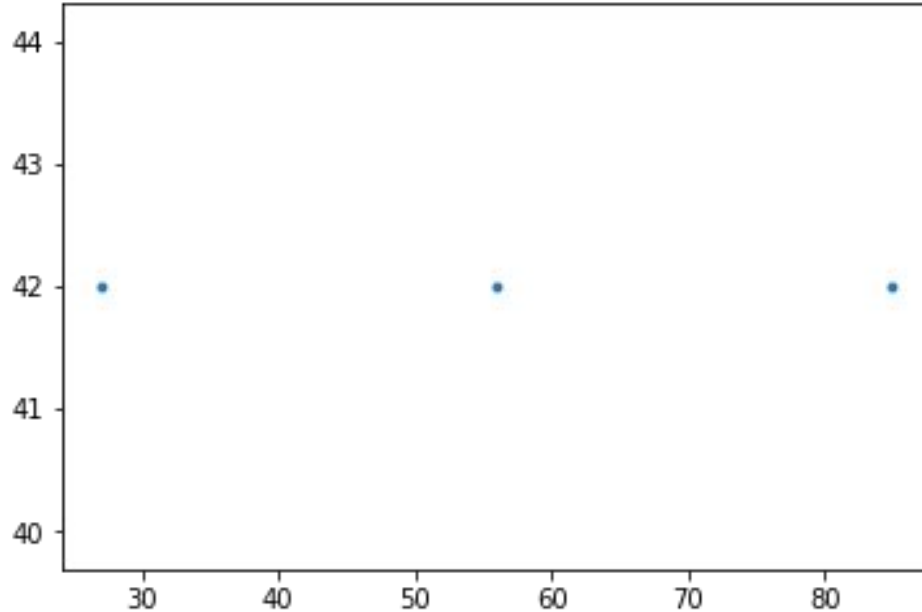
```
plt.plot(v[0].times, v[0])
```

```
plt.show()
```



PyNN - Plotting

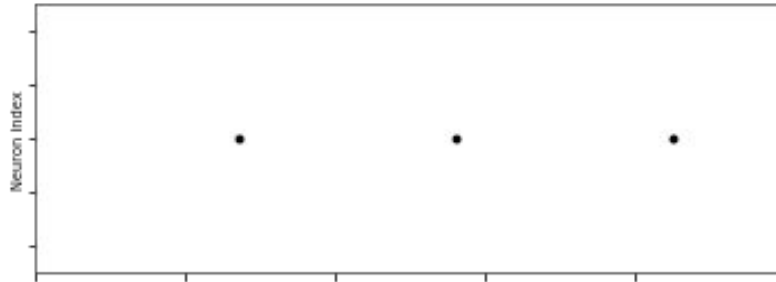
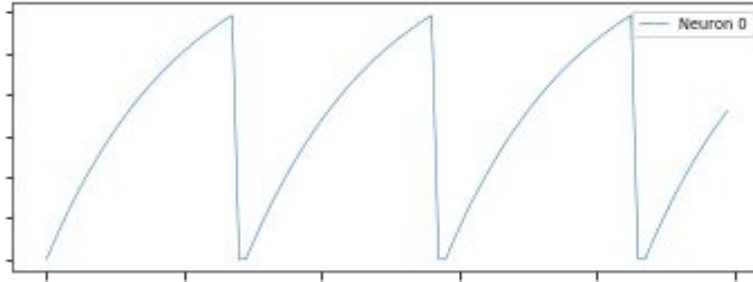
```
plt.figure()  
y = [1 for i in range(len(spikes[0]))]  
plt.plot(spikes[0], y, ".")  
plt.show()
```



PyNN - Plotting Neo Directly

```
from pyNN.utility.plotting import Figure, Panel
Figure(
```

```
    Panel(*data.segments[0].analogsignals),
    Panel(data.segments[0].spiketrains)
)
```



PyNN - Get Weights and Delays

```
synapses = proj.get(
    ["weight", "delay"], "list")

array([(0, 5, 0.756, 1.), (0, 6, 0.316, 1.), (0, 7, 0.885, 2.),
       (0, 8, 0.421, 1.), (1, 4, 0.618, 1.), (1, 7, 0.438, 1.),
       (1, 9, 1.607, 1.), (2, 0, 0.129, 1.), (2, 2, 1.055, 1.),
       (2, 3, 1.319, 1.), (2, 9, 0.422, 1.), (3, 1, 0.328, 1.),
       (3, 3, 0.456, 1.), (3, 6, 0.566, 1.), (4, 0, 1.046, 1.),
       (4, 1, 1.199, 1.), (4, 2, 0.831, 1.), (5, 0, 1.643, 1.),
       (5, 2, 1.165, 1.), (5, 3, 0.902, 1.), (5, 5, 1.627, 1.),
       (6, 0, 2.143, 1.), (6, 5, 0.635, 1.), (6, 7, 0.704, 1.),
       (7, 0, 1.914, 1.), (7, 4, 0.289, 1.), (7, 5, 2.058, 1.),
       (7, 6, 0.428, 2.), (7, 7, 0.639, 1.), (7, 9, 0.616, 2.),
       (8, 0, 1.039, 1.), (8, 1, 0.576, 1.), (8, 4, 1.563, 2.),
       (8, 8, 0.995, 1.), (9, 0, 1.686, 1.), (9, 9, 0.631, 2.)])
```

PyNN - Get Weights and Delays

```
synapses = proj.get(  
    "weight", "array")
```

```
array([[ nan,  nan,  nan,  nan,  nan,  0.756,  0.316,  0.885,  0.421,  nan],  
       [ nan,  nan,  nan,  nan,  0.618,  nan,  nan,  0.438,  nan,  1.607],  
       [0.129,  nan,  1.055,  1.319,  nan,  nan,  nan,  nan,  nan,  0.422],  
       [ nan,  0.328,  nan,  0.456,  nan,  nan,  0.566,  nan,  nan,  nan],  
       [1.046,  1.199,  0.831,  nan,  nan,  nan,  nan,  nan,  nan,  nan],  
       [1.643,  nan,  1.165,  0.902,  nan,  1.627,  nan,  nan,  nan,  nan],  
       [2.143,  nan,  nan,  nan,  nan,  0.635,  nan,  0.704,  nan,  nan],  
       [1.914,  nan,  nan,  nan,  0.289,  2.058,  0.428,  0.639,  nan,  0.616],  
       [1.039,  0.576,  nan,  nan,  1.563,  nan,  nan,  nan,  0.995,  nan],  
       [1.686,  nan,  nan,  nan,  nan,  nan,  nan,  nan,  nan,  0.631]])
```

<http://neuralensemble.org/docs/PyNN/>

PyNN 0.9.4 documentation »

[next](#) | [modules](#) | [index](#)



Table of Contents

- PyNN: documentation
 - Developers' Guide
 - API reference
 - Old documents
 - Indices and tables

Next topic

[Introduction](#)

This Page

[Show Source](#)

Quick search

Go

PyNN: documentation

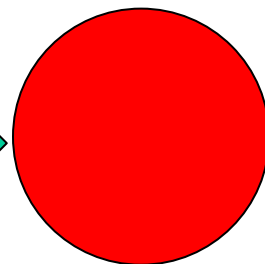
- [Introduction](#)
- [Installation](#)
- [Quickstart](#)
- [Building networks](#)
- [Injecting current](#)
- [Recording spikes and state variables](#)
- [Data handling](#)
- [Simulation control](#)
- [Model parameters and initial values](#)
- [Random numbers](#)
- [Backends](#)
- [Running parallel simulations](#)
- [Units](#)
- [Importing from and exporting to other formats](#)
- [Examples](#)
- [Publications about, relating to or using PyNN](#)
- [Contributors and licence](#)
- [Release notes](#)

Live Input and Output

Input from Environment: Spikes

send_spikes

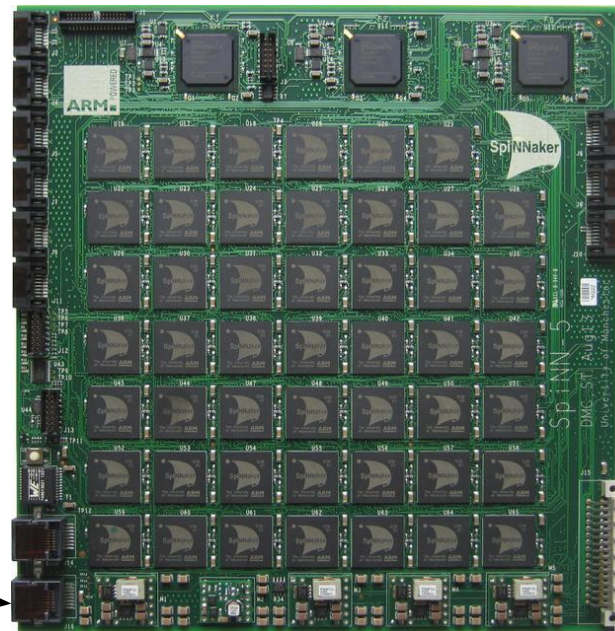
SpikeInjector
label="injector"



send_spikes("injector", [0])

```
SpynnakerLiveSpikesConnection(  
    send_labels=["injector"])
```

Multicast Key(s)
(Spike)



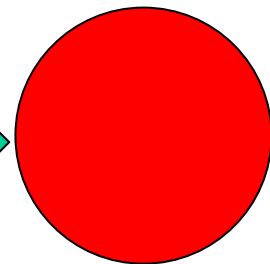
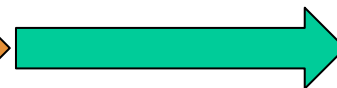
Input from Environment: Rates

add_poisson_live_rate_control



set_rates

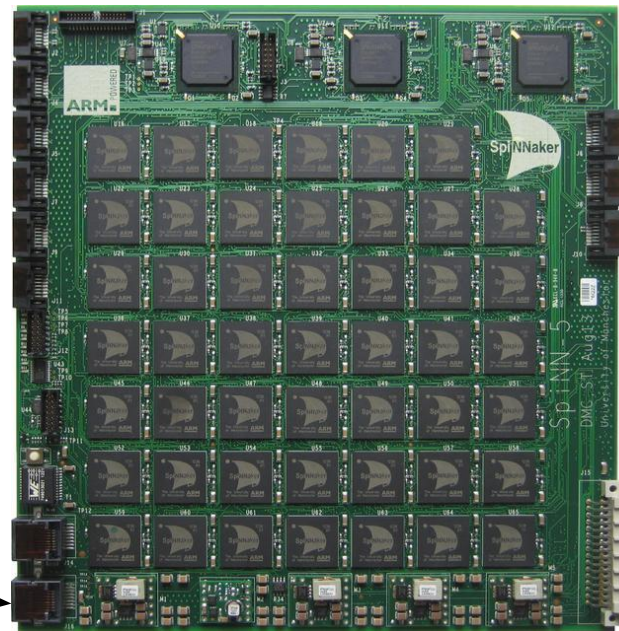
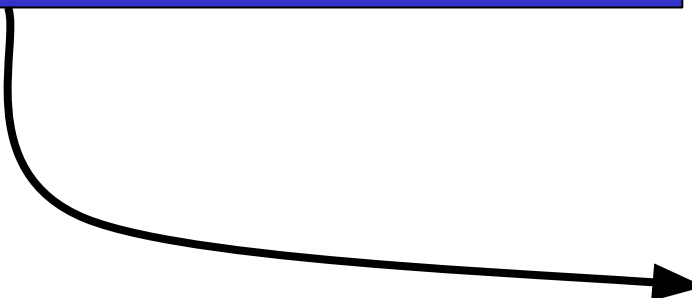
SpikeSourcePoisson
label="input"



set_rates("input", [(0, 10)])

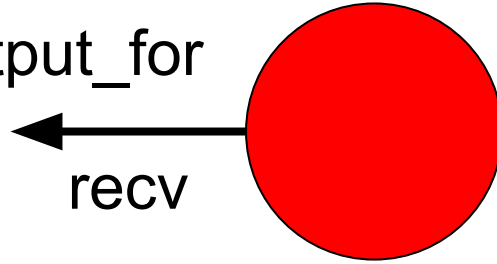
SpynakerPoissonControlConnection
poisson_labels=["input"]

Multicast Key(s)
and Payload(s)



Output to Environment: Spikes

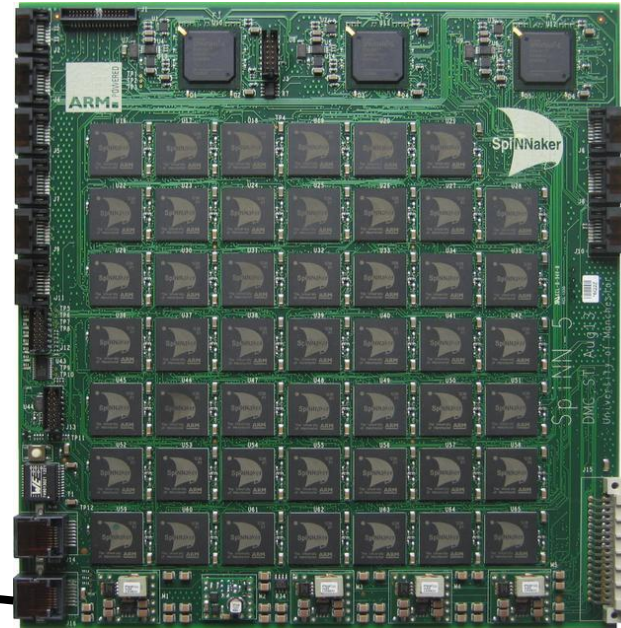
activate_live_output_for



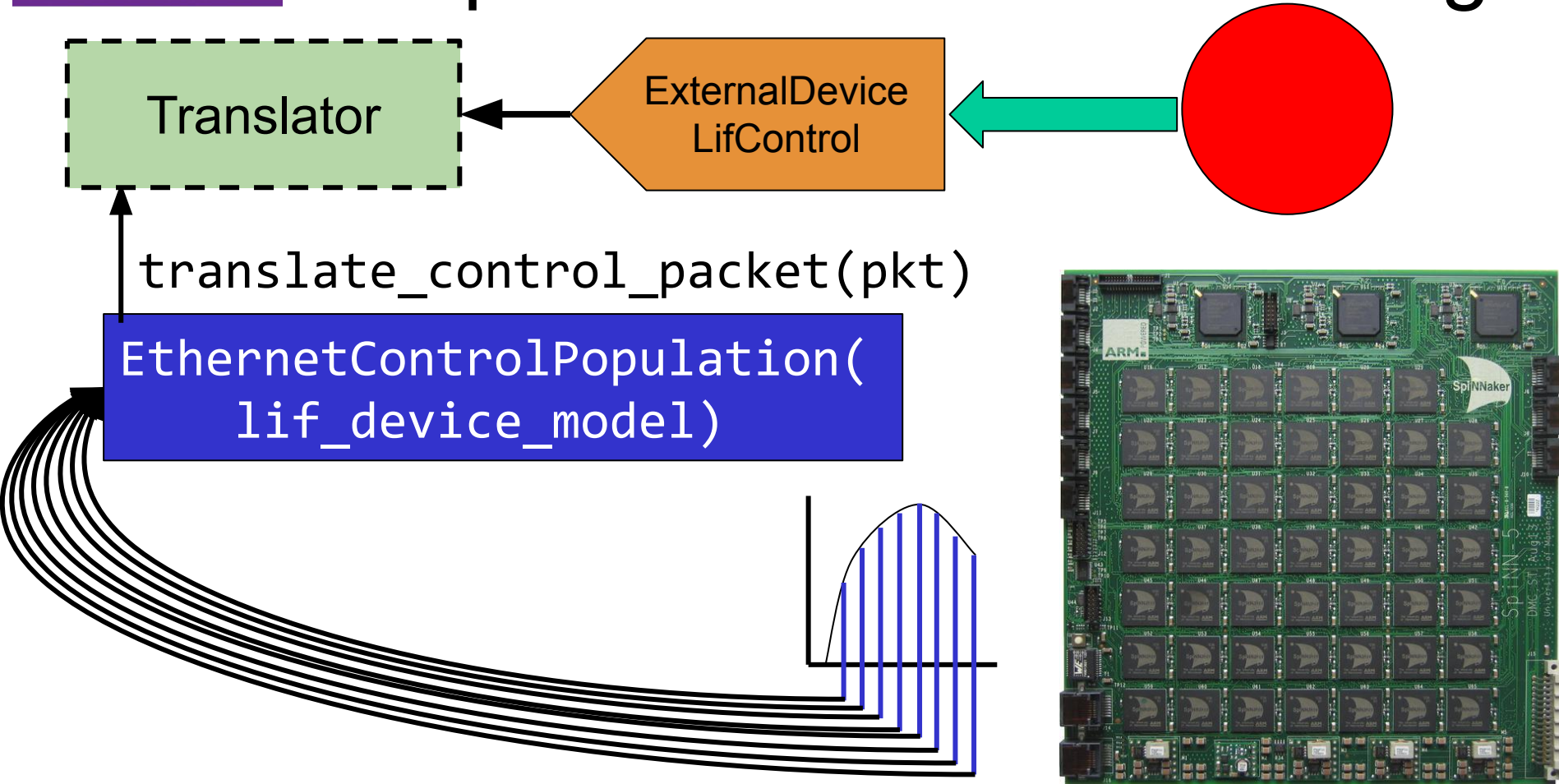
recv("pop", time, neuron_ids)

```
SpynakerLiveSpikesConnection(  
    receive_labels=["pop"])
```

Multicast Key(s)
(Spike)



Output to Environment: Voltage



SpiNNaker Tutorials on Jupyter

<https://spinn-20.cs.man.ac.uk/>

01. Running PyNN Simulations

02. Live Input And Output

PyNN Documentation:

<http://neuralensemble.org/docs/PyNN/>

For NRP: New → Terminal, then run `cle-nginx`, then `cle-start`

<https://spinn-20.cs.man.ac.uk/user/<username>/proxy/9000/#/esv-private>