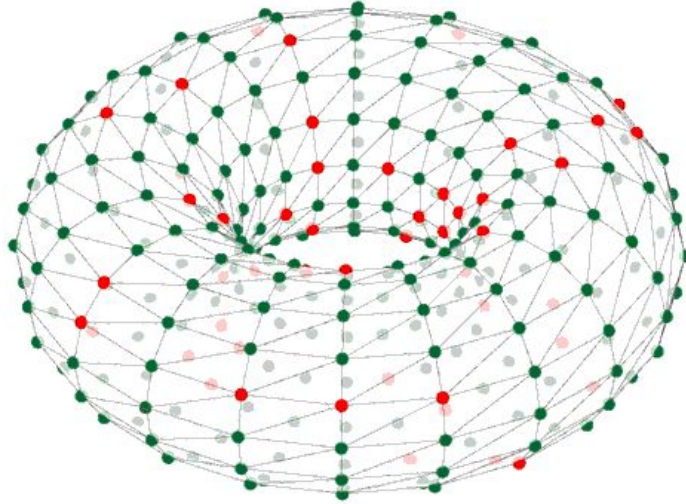


SpiNNaker Tutorial

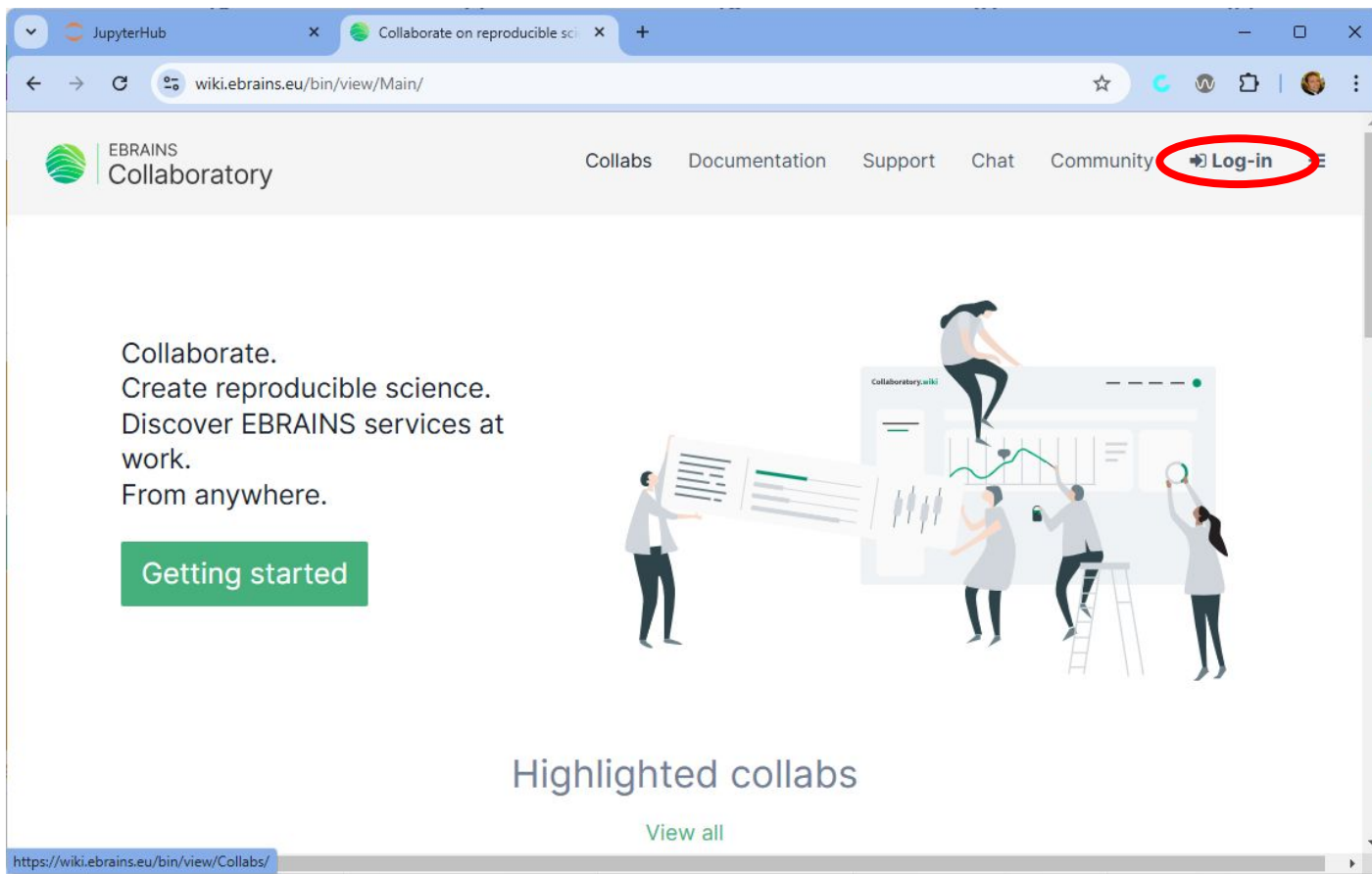


Accessing SpiNNaker

Accessing SpiNNaker via Jupyter - Create a Collaboratory

<https://wiki.ebrains.eu/>

Log in to Wiki



The screenshot shows a web browser window with two tabs: 'JupyterHub' and 'Collaborate on reproducible sci...'. The address bar shows the URL 'wiki.ebrains.eu/bin/view/Main/'. The page header includes the 'EBRAINS Collaboratory' logo and a navigation menu with links for 'Collabs', 'Documentation', 'Support', 'Chat', 'Community', and 'Log-in'. The 'Log-in' link is circled in red. The main content area features a large illustration of four people collaborating around a large screen displaying a line graph. To the left of the illustration, the text reads: 'Collaborate. Create reproducible science. Discover EBRAINS services at work. From anywhere.' Below this text is a green button labeled 'Getting started'. At the bottom of the page, the text 'Highlighted collabs' is displayed, followed by a green link 'View all'. The browser's status bar at the bottom shows the URL 'https://wiki.ebrains.eu/bin/view/Collabs/'.

JupyterHub

Collaborate on reproducible sci...

wiki.ebrains.eu/bin/view/Main/

EBRAINS Collaboratory

Collabs Documentation Support Chat Community **Log-in**

Collaborate.
Create reproducible science.
Discover EBRAINS services at work.
From anywhere.

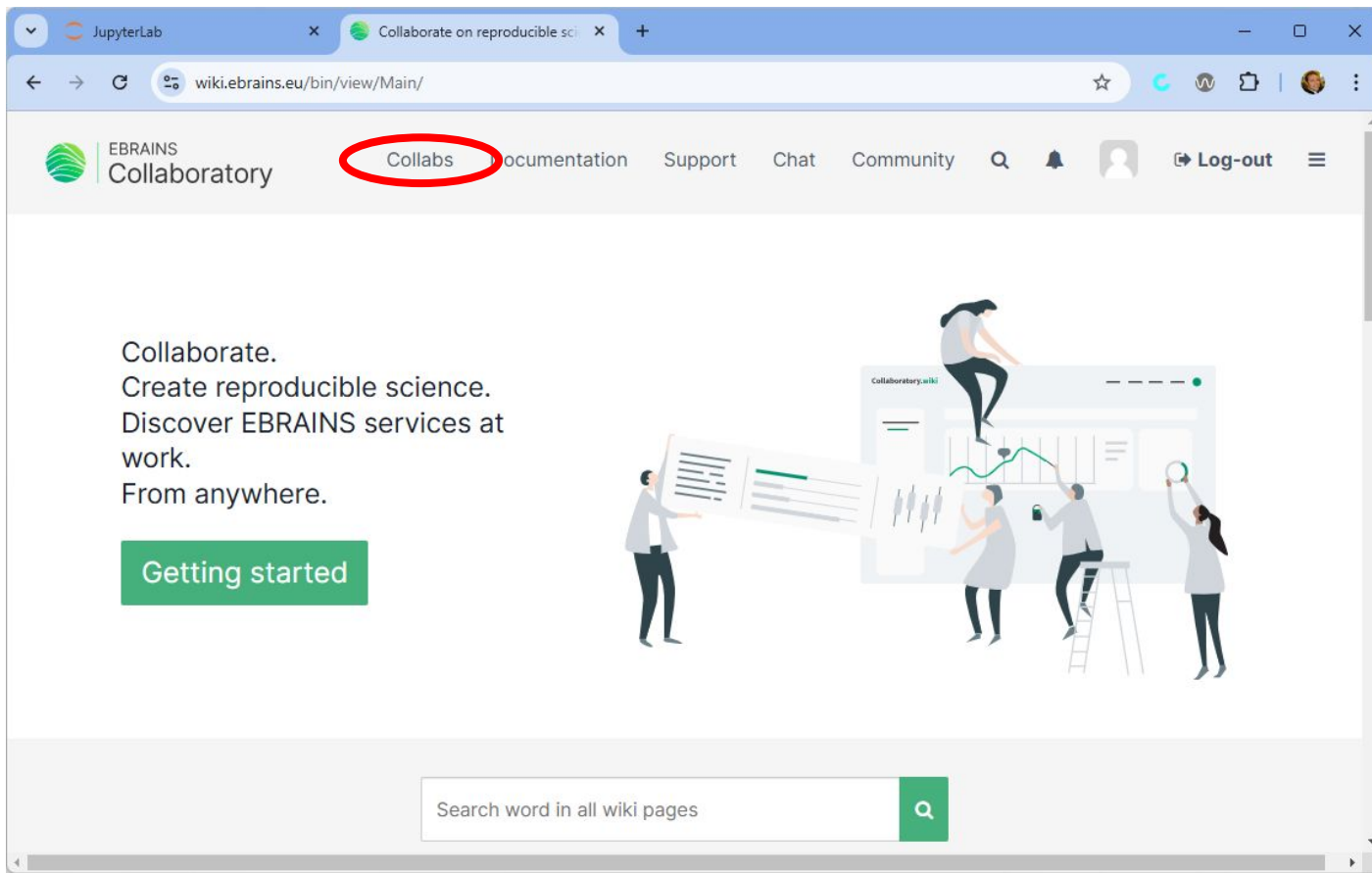
Getting started

Highlighted collabs

[View all](#)

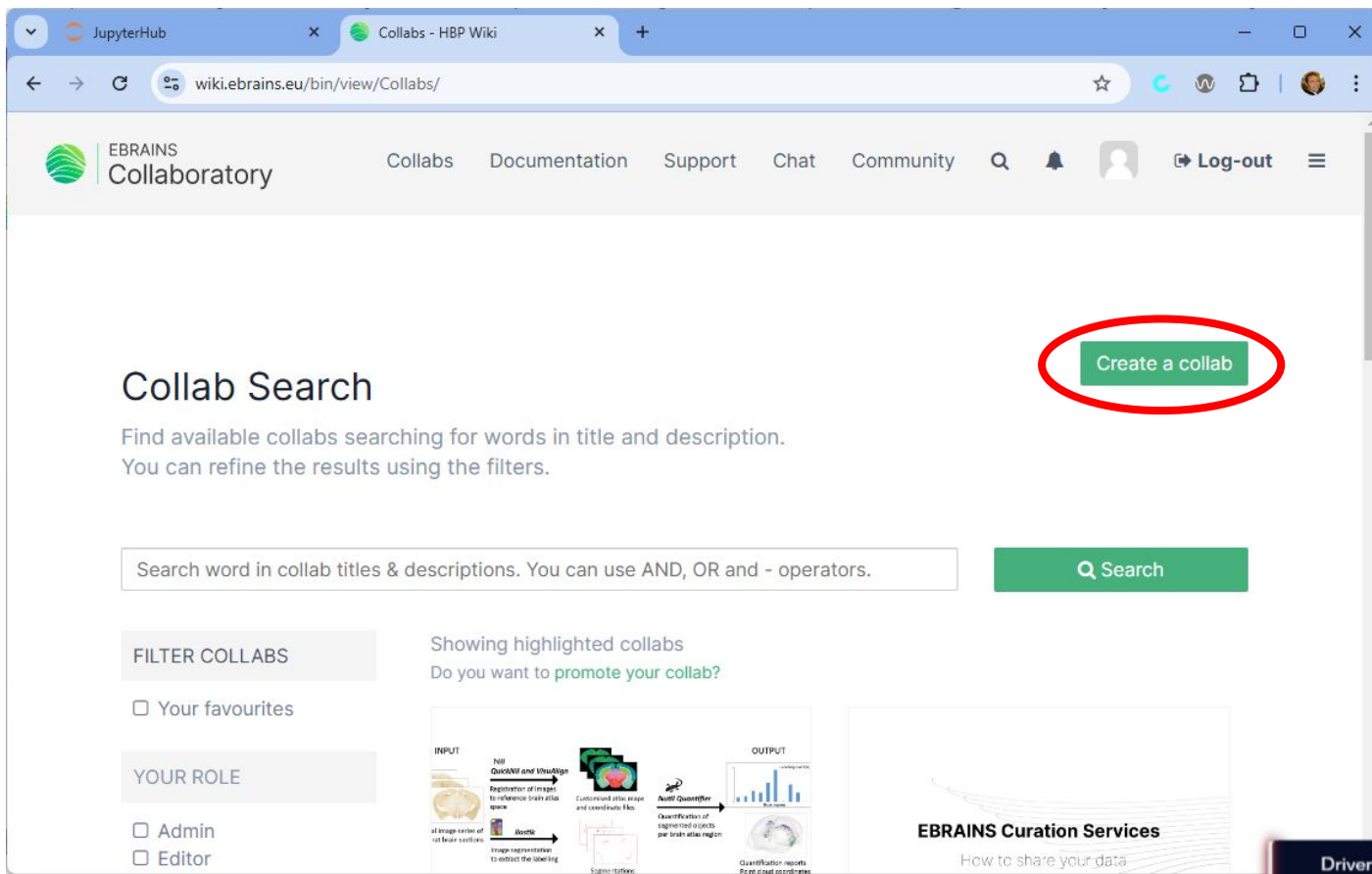
https://wiki.ebrains.eu/bin/view/Collabs/

Create a Collaboratory



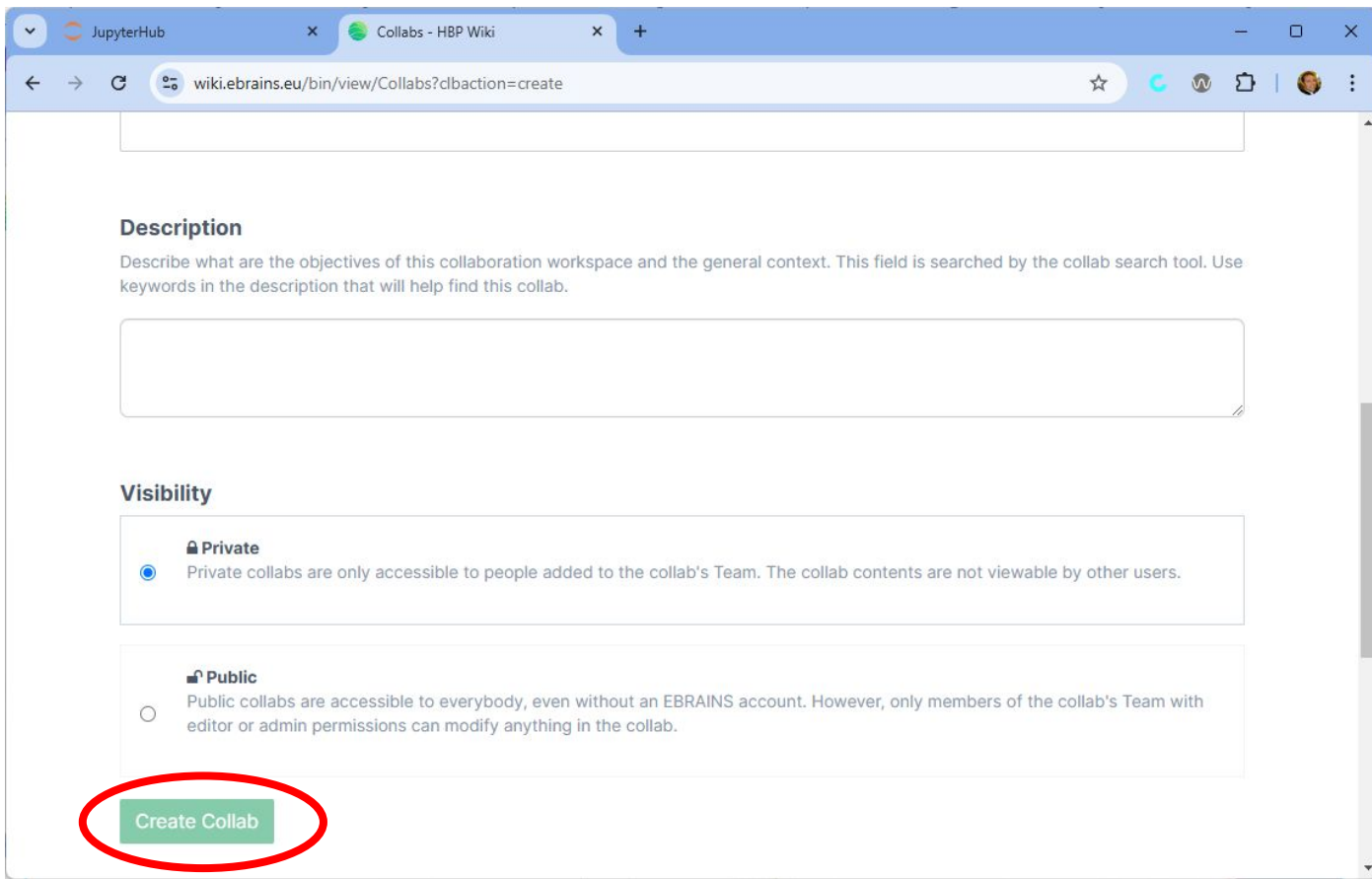
The screenshot shows a web browser window with two tabs: 'JupyterLab' and 'Collaborate on reproducible sci...'. The address bar shows the URL 'wiki.ebrains.eu/bin/view/Main/'. The website header features the 'EBRAINS Collaboratory' logo on the left and a navigation bar on the right. The 'Collabs' link in the navigation bar is circled in red. Other navigation links include 'Documentation', 'Support', 'Chat', 'Community', a search icon, a notification bell, a user profile icon, and a 'Log-out' button. The main content area has the text: 'Collaborate. Create reproducible science. Discover EBRAINS services at work. From anywhere.' Below this text is a green button labeled 'Getting started'. To the right of the text is an illustration of four people interacting with a large digital screen displaying a line graph and data. At the bottom of the page is a search bar with the placeholder text 'Search word in all wiki pages' and a green search button.

Create a Collaboratory



The screenshot shows a web browser window with two tabs: 'JupyterHub' and 'Collabs - HBP Wiki'. The address bar shows 'wiki.ebrains.eu/bin/view/Collabs/'. The page header includes the 'EBRAINS Collaboratory' logo and navigation links: 'Collabs', 'Documentation', 'Support', 'Chat', 'Community', a search icon, a bell icon, a user profile icon, and a 'Log-out' button. The main content area is titled 'Collab Search' and includes the text: 'Find available collabs searching for words in title and description. You can refine the results using the filters.' Below this is a search input field with the placeholder text 'Search word in collab titles & descriptions. You can use AND, OR and - operators.' and a green 'Search' button. On the left side, there are two filter sections: 'FILTER COLLABS' with a checkbox for 'Your favourites', and 'YOUR ROLE' with checkboxes for 'Admin' and 'Editor'. On the right side, there is a green button labeled 'Create a collab' which is circled in red. Below the search bar, there is a section titled 'Showing highlighted collabs' with the text 'Do you want to promote your collab?'. At the bottom, there is a diagram showing the workflow from 'INPUT' (NIfTI QuickNII and Virtualization, Registration of images to reference brain atlas space, Customized atlas shape and coordinate files, Avicore, Image representation to extract the labeling, Segmentation) to 'OUTPUT' (Nifty Quantifier, Quantification of segmented subjects per brain atlas region, Quantification reports, Brain cloud coordinates). To the right of the diagram is a section titled 'EBRAINS Curation Services' with the text 'How to share your data'.

Create a Collaboratory



JupyterHub Collabs - HBP Wiki

wiki.ebrains.eu/bin/view/Collabs?dbaction=create

Description

Describe what are the objectives of this collaboration workspace and the general context. This field is searched by the collab search tool. Use keywords in the description that will help find this collab.

Visibility

☒ **Private**
Private collabs are only accessible to people added to the collab's Team. The collab contents are not viewable by other users.

☐ **Public**
Public collabs are accessible to everybody, even without an EBRAINS account. However, only members of the collab's Team with editor or admin permissions can modify anything in the collab.

Create Collab

Accessing SpiNNaker via Jupyter - Go to EBRAINS Lab

<https://lab.ebrains.eu/>

EBRAINS Lab - Choose a Site

Jupyterlab - Execution Site

lab.ebrains.eu

jupyter

Select Lab Execution Site ?

☒ Fenix CH - Swiss National Supercomputing Center (CSCS)

☐ Fenix DE - Jülich Supercomputing Center (JSC)

☐ Fenix ES - Barcelona Supercomputing Center (BSC)

☐ Fenix FR - French Alternative Energies and Atomic Energy Commission (CEA)

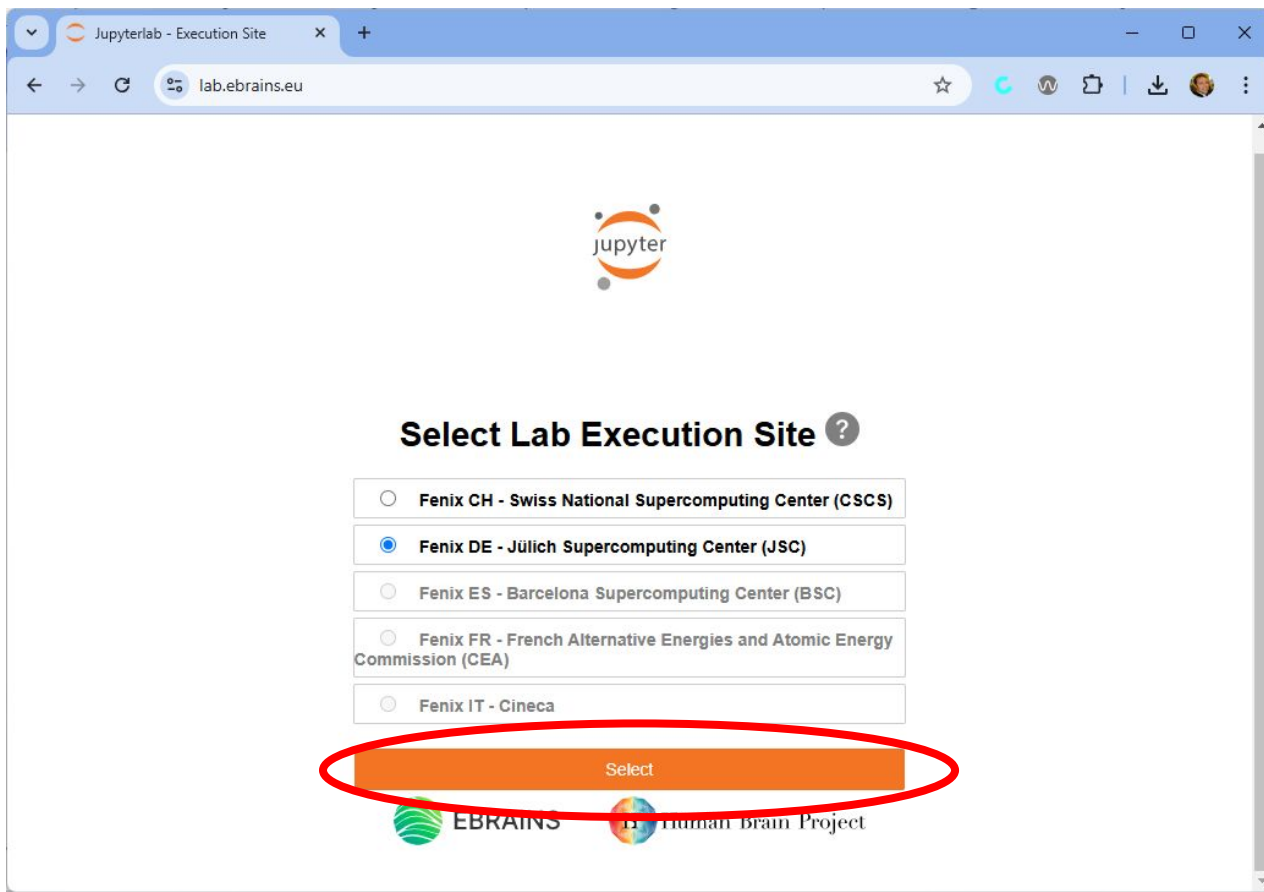
☐ Fenix IT - Cineca

Select

EBRAINS

Human Brain Project

EBRAINS Lab - Choose a Site



Jupyterlab - Execution Site

lab.ebrains.eu

jupyter

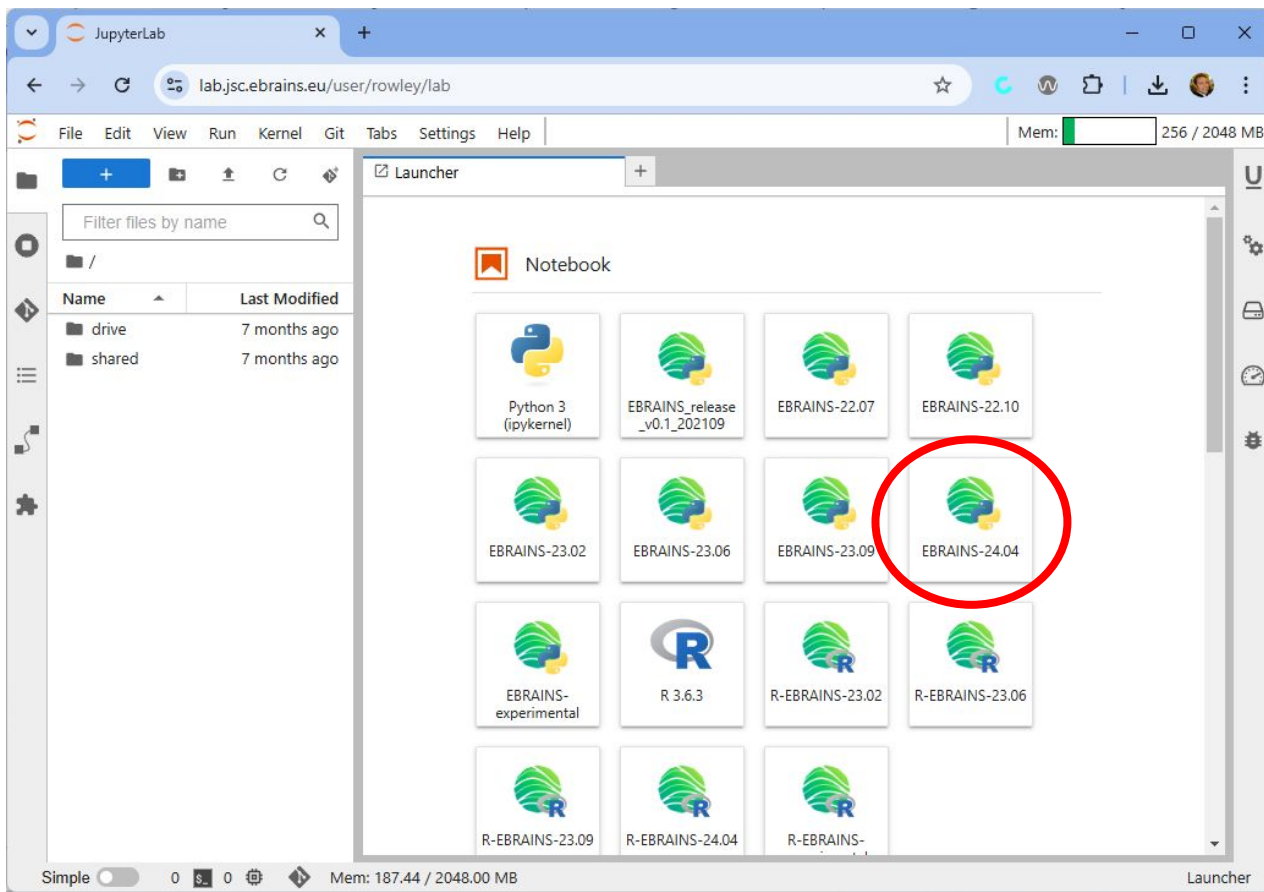
Select Lab Execution Site ?

- ☐ Fenix CH - Swiss National Supercomputing Center (CSCS)
- ☒ Fenix DE - Jülich Supercomputing Center (JSC)
- ☐ Fenix ES - Barcelona Supercomputing Center (BSC)
- ☐ Fenix FR - French Alternative Energies and Atomic Energy Commission (CEA)
- ☐ Fenix IT - Cineca

Select

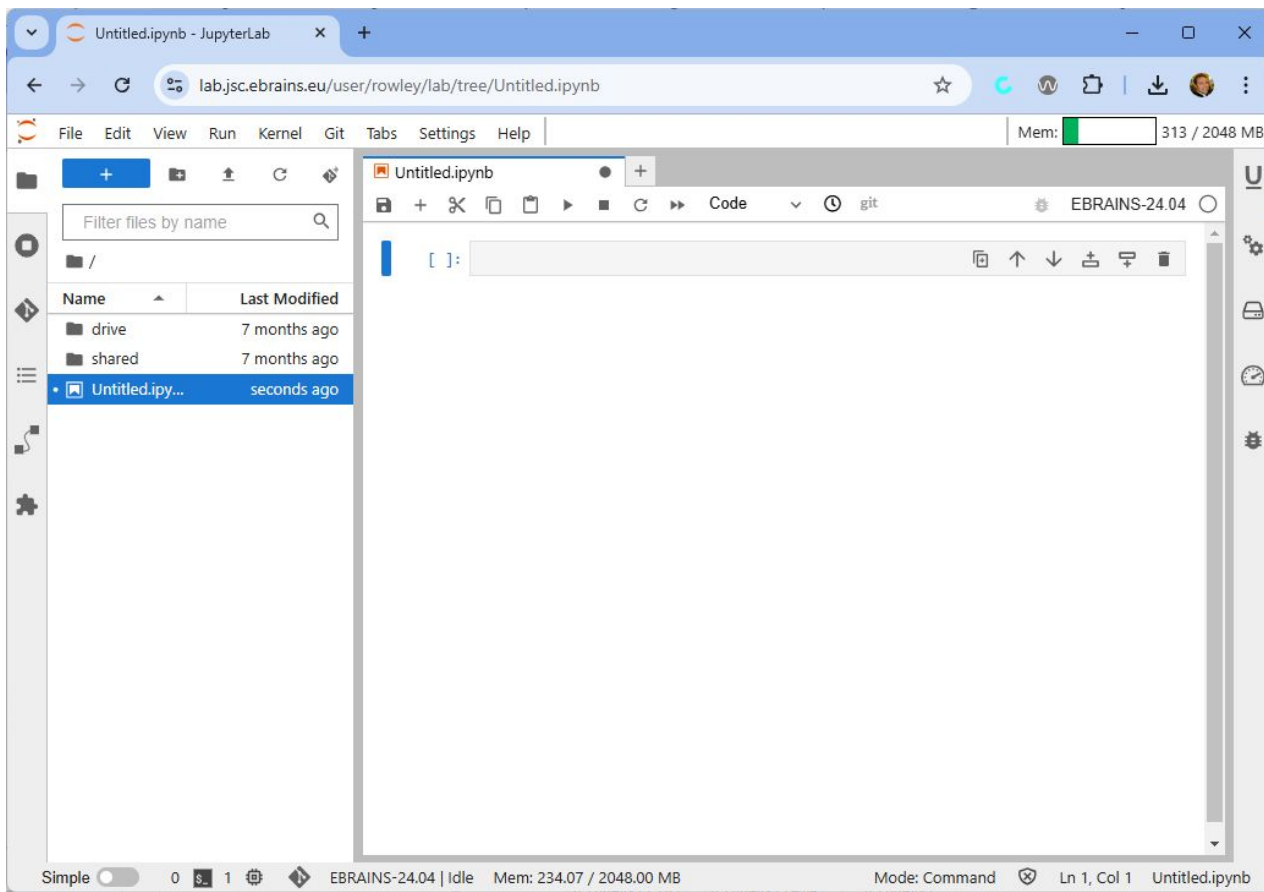
EBRAINS Human Brain Project

EBRAINS Lab - Choose a Kernel



The screenshot shows the JupyterLab web interface in a browser. The address bar displays `lab.jsc.ebrains.eu/user/rowley/lab`. The top navigation bar includes menus for File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. On the right of this bar, a memory usage indicator shows 'Mem: 256 / 2048 MB'. The left sidebar contains a file browser with a search bar and a list of files and folders. The main area is titled 'Launcher' and displays a grid of available kernels. A red circle highlights the 'EBRAINS-24.04' kernel, which is the selected option. The kernels are organized into rows: the first row contains 'Python 3 (ipykernel)', 'EBRAINS_release_v0.1_202109', 'EBRAINS-22.07', and 'EBRAINS-22.10'; the second row contains 'EBRAINS-23.02', 'EBRAINS-23.06', 'EBRAINS-23.09', and 'EBRAINS-24.04'; the third row contains 'EBRAINS-experimental', 'R 3.6.3', 'R-EBRAINS-23.02', and 'R-EBRAINS-23.06'; and the fourth row contains 'R-EBRAINS-23.09', 'R-EBRAINS-24.04', and 'R-EBRAINS-24.06'. The bottom status bar shows 'Simple' as the selected theme and 'Mem: 187.44 / 2048.00 MB'.

EBRAINS Lab - Choose a Kernel



Find your Collab: /shared/...

The screenshot shows the JupyterLab interface with two main panels: the File Browser on the left and the Launcher on the right.

File Browser: The left panel displays a list of files and folders. The folder `/ shared /` is highlighted with a red circle. Below it, a table lists various files and folders with their last modified dates.

Name	Last Modified
Andrew Public Test	a year ago
Andrew Rowley Default Quota	13 days ago
Andrew Rowley Default Quota 2	13 days ago
Andrew Test Collab	2 days ago
Andrew's Testing Collab	5 years ago
AngoraPy	2 years ago
API Catalogue	3 years ago
Async-Neuromorph	a year ago
Atlas pipeline workflow	a year ago
Atlas registration of neuronal mor...	a year ago
Atlas Viewer Development	3 years ago
Basal Ganglia Mean Field	2 years ago
BASSES Hands-on Session I - Han...	2 years ago
BASSES Hands-on Session II - Run...	2 years ago
BASSES Hands-on Session III - Sim...	2 years ago
BASSES Workshop Rome Event M...	2 years ago
Bayesian Virtual Epileptic Patient	2 years ago
BCCN-TVb-workshop-2022	2 years ago

Launcher: The right panel shows a grid of notebooks. The top row includes `Python 3 (ipykernel)` and three EBRAINS notebooks: `EBRAINS_release_v0.1_202109`, `EBRAINS-22.07`, and `EBRAINS-22.10`. The second row includes `EBRAINS-23.02`, `EBRAINS-23.06`, `EBRAINS-23.09`, and `EBRAINS-24.04`. The third row includes `EBRAINS-experimental`, `R 3.6.3`, `R-EBRAINS-23.02`, and `R-EBRAINS-23.06`.

Clone SpiNNaker Repository

The screenshot displays the JupyterLab web interface. The top navigation bar includes tabs for 'JupyterLab' and 'Collaborate on reproducible sci...'. The address bar shows the URL 'lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab'. The top menu bar contains 'File', 'Edit', 'View', 'Run', 'Kernel', 'Git', 'Tabs', 'Settings', and 'Help'. The 'Git' icon is circled in red. Below the menu, the left sidebar shows a file browser with a search bar and a list of files and folders. The right sidebar shows the 'Launcher' view with a grid of notebooks.

File Browser (Left Sidebar):

Name	Last Modified
reports	3 months ago
test	a year ago
SpiNNaker_01_test-Copy1.ipynb	a year ago
SpiNNaker_01_test.ipynb	a year ago
SpiNNaker_Collab_synfire.ipynb	3 months ago
spynnaker.cfg	a year ago
Test.ipynb	3 months ago
Untitled.ipynb	a year ago

Launcher (Right Sidebar):

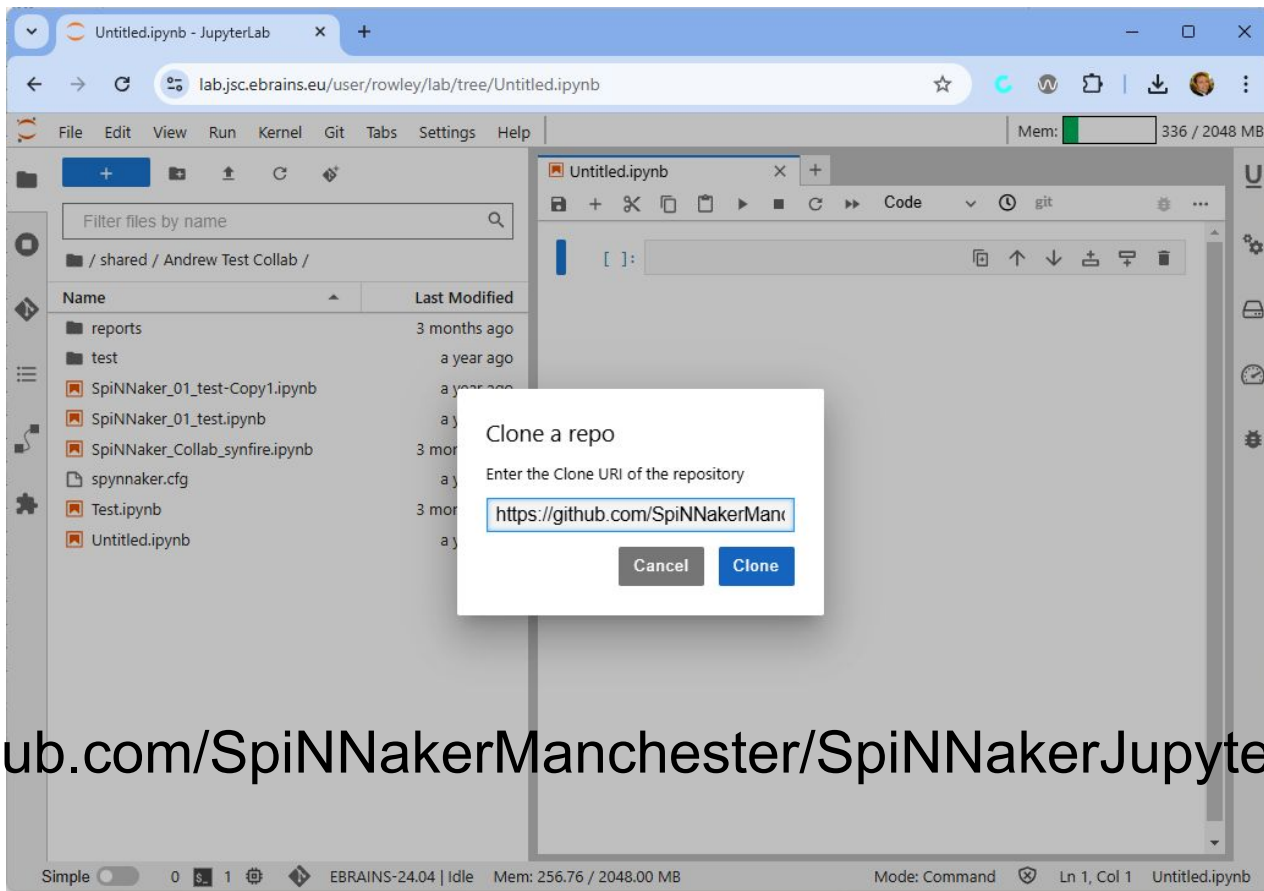
shared/Andrew Test Collab

Notebook

Python 3 (ipykernel)	EBRAINS_release_v0.1_202109	EBRAINS-22.07	EBRAINS-22.10
EBRAINS-23.02	EBRAINS-23.06	EBRAINS-23.09	EBRAINS-24.04
EBRAINS-experimental	R 3.6.3	R-EBRAINS-23.02	R-EBRAINS-23.06

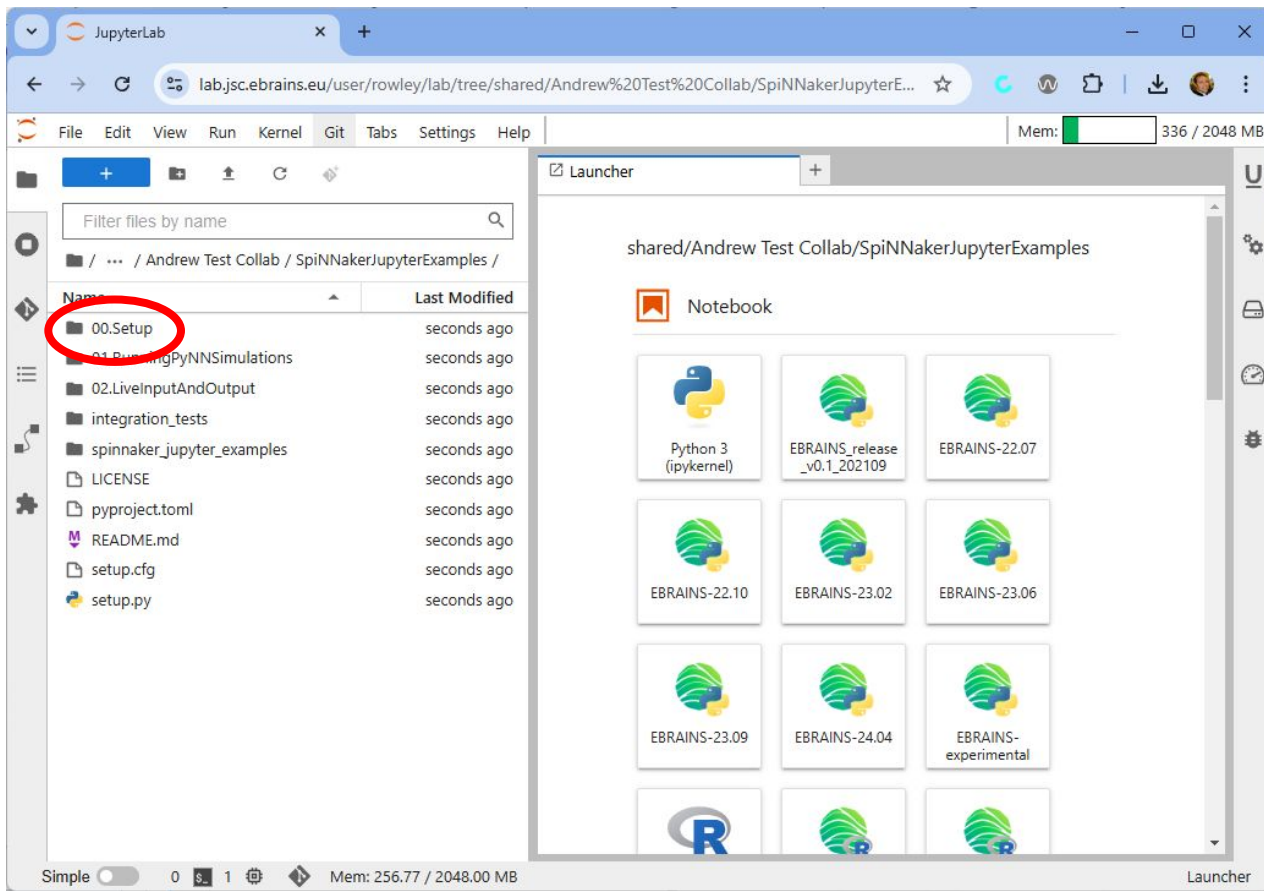
Simple 1 0 Mem: 260.07 / 2048.00 MB Launcher

Clone SpiNNaker Repository



<https://github.com/SpiNNakerManchester/SpiNNakerJupyterExamples>

Clone SpiNNaker Repository



The screenshot shows a JupyterLab web interface in a browser. The address bar displays the URL: `lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterE...`. The interface includes a top menu bar with options like File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. Below the menu is a toolbar with icons for file operations. The left sidebar contains a file explorer with a search bar and a list of files and folders. The file `00.Setup` is highlighted with a red circle. The right sidebar shows a launcher panel with a grid of notebook icons, including Python 3 (ipykernel) and various EBRAINS releases.

File Explorer (Left Panel):

Name	Last Modified
00.Setup	seconds ago
01.RunningPyNNSimulations	seconds ago
02.LiveInputAndOutput	seconds ago
integration_tests	seconds ago
spinnaker_jupyter_examples	seconds ago
LICENSE	seconds ago
pyproject.toml	seconds ago
README.md	seconds ago
setup.cfg	seconds ago
setup.py	seconds ago

Launcher Panel (Right Panel):

shared/Andrew Test Collab/SpiNNakerJupyterExamples

Notebook

Python 3 (ipykernel)	EBRAINS_release_v0.1_202109	EBRAINS-22.07
EBRAINS-22.10	EBRAINS-23.02	EBRAINS-23.06
EBRAINS-23.09	EBRAINS-24.04	EBRAINS-experimental
R		

At the bottom of the interface, the status bar shows "Simple" mode, a memory usage of 256.77 / 2048.00 MB, and a "Launcher" button.

Open Setup Notebook

The screenshot displays the JupyterLab interface. The browser tab is titled 'Setup.ipynb - JupyterLab'. The address bar shows the URL 'lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterE...'. The interface includes a menu bar (File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help) and a toolbar. The left sidebar shows a file explorer with a search bar and a list of files, including 'Setup.ipynb'. The main area displays the 'Setup.ipynb' notebook. A 'Select Kernel' dialog box is open, asking to 'Select kernel for: "Setup.ipynb"'. The dropdown menu shows 'Python 3 (ipykernel)' as the selected option. There are 'No Kernel' and 'Select' buttons. The notebook content includes a section titled 'Jupyter Setup' and a code block for a function 'def check_should_create()'.

Launcher

Setup.ipynb

Markdown

git

No Kernel

Filter files by name

Name

Last Modified

Setup.ipynb

seconds ago

Select Kernel

Select kernel for: "Setup.ipynb"

Python 3 (ipykernel)

No Kernel

Select

Jupyter Setup

The code below is needed to create a sPyNNaker configuration file, which is needed on the first run of the software. It will be necessary to start the Lab (it can also be added as the kernel if desired). A future image may already have the need to do this.

```
def check_should_create():
    """
    Detects if there is an existing cfg file and if so if it
    :return:
    """
    if not os.path.isfile(FULL_PATH):
        return True
    if OVERRIDE_EXISTING:
        return True
```

Simple

0

1

No Kernel | Initializing

Mem: 256.77 / 2048.00 MB

Mode: Command

Ln 1, Col 1

Setup.ipynb

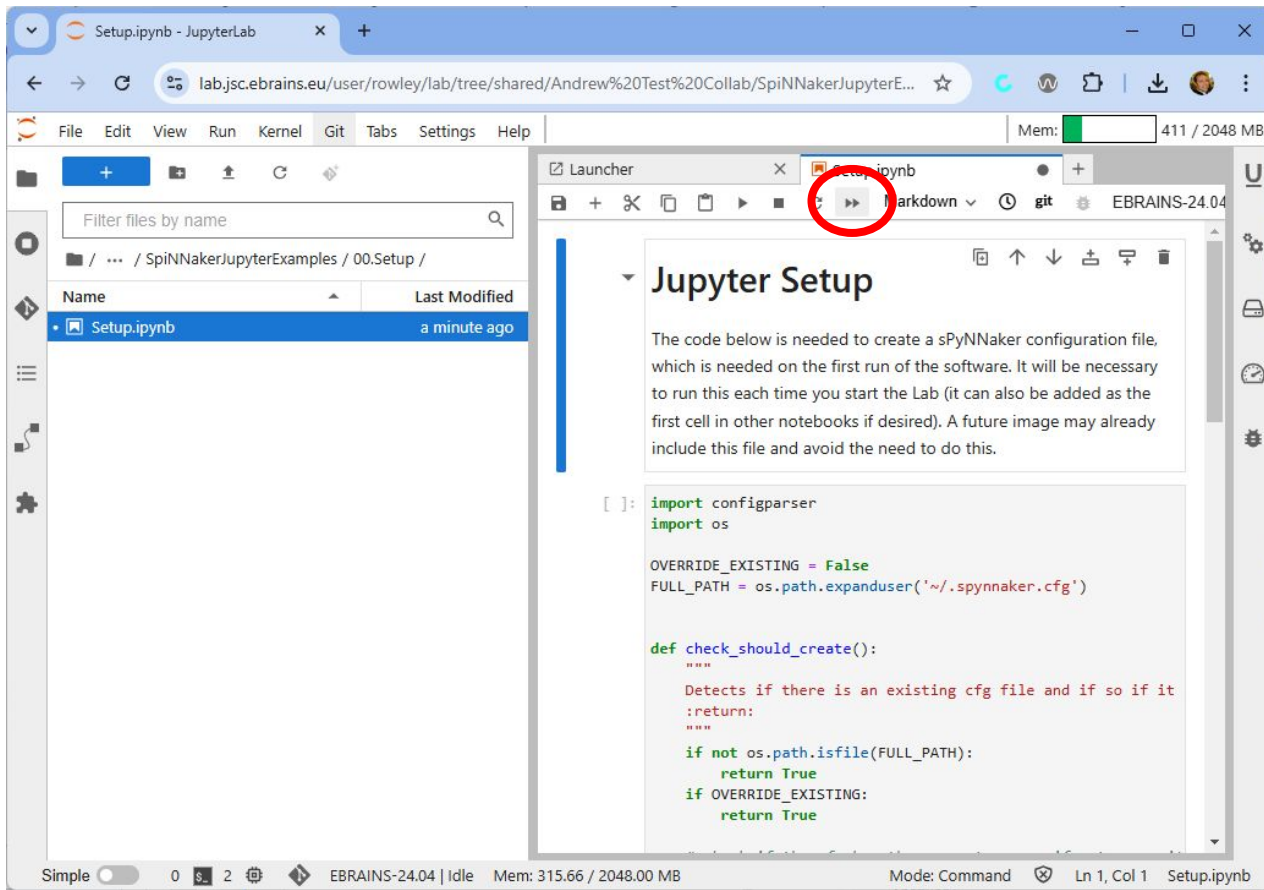
Open Setup Notebook

The screenshot shows a JupyterLab environment with the following details:

- Browser Tab:** Setup.ipynb - JupyterLab
- URL:** lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterE...
- File Explorer:** Shows the file structure with 'Setup.ipynb' selected in the '00.Setup' directory.
- Select Kernel Dialog:** A modal window titled 'Select Kernel' is open, showing 'EBRAINS-24.04' as the selected kernel for 'Setup.ipynb'. It has 'No Kernel' and 'Select' buttons.
- Notebook Content:**
 - Title:** Jupyter Setup
 - Text:** The code below is needed to create a sPyNNaker configuration file, which is needed on the first run of the software. It will be necessary to start the Lab (it can also be added as the kernel if desired). A future image may already have the need to do this.
 - Code:**

```
def check_should_create():  
    """  
    Detects if there is an existing cfg file and if so if it  
    :return:  
    """  
    if not os.path.isfile(FULL_PATH):  
        return True  
    if OVERRIDE_EXISTING:  
        return True
```
- Status Bar:** Shows 'Simple' mode, 'No Kernel | Initializing', memory usage 'Mem: 257.83 / 2048.00 MB', and 'Mode: Command'.

Open Setup Notebook - Run



The screenshot shows a JupyterLab interface with a browser window at the top. The address bar shows the URL: `lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterE...`. The interface includes a menu bar (File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help) and a toolbar with various icons. A file browser on the left shows the directory structure: `/ SpiNNakerJupyterExamples / 00.Setup /`, with `Setup.ipynb` selected. The main area displays the notebook content, which includes a title `Jupyter Setup`, a paragraph explaining the need for a configuration file, and a code cell with the following Python code:

```
[ ]: import configparser
import os

OVERRIDE_EXISTING = False
FULL_PATH = os.path.expanduser('~/.spynaker.cfg')

def check_should_create():
    """
    Detects if there is an existing cfg file and if so if it
    :return:
    """
    if not os.path.isfile(FULL_PATH):
        return True
    if OVERRIDE_EXISTING:
        return True
```

A red circle highlights the 'Run' button (a double right arrow icon) in the toolbar above the code cell.

Open Setup Notebook - Run

The screenshot displays the JupyterLab web interface in a browser. The browser's address bar shows the URL: `lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterE...`. The JupyterLab interface includes a top menu bar (File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help) and a memory usage indicator (411 / 2048 MB). On the left, a file browser shows the directory structure: `/ ... / SpiNNakerJupyterExamples / 00.Setup /`, with `Setup.ipynb` selected. The main editor area shows the `Setup.ipynb` notebook, which has a title bar with `Launcher` and `Setup.ipynb`. The notebook content includes a heading `Jupyter Setup` and a paragraph explaining the need for a configuration file. Below the text, there is a code cell containing Python code for checking and creating a configuration file. A modal dialog box is centered on the screen, asking 'Restart Kernel?' with the message 'Do you want to restart the current kernel? All variables will be lost.' and two buttons: 'Cancel' and 'Restart'.

Restart Kernel?

Do you want to restart the current kernel? All variables will be lost.

Cancel Restart

Jupyter Setup

The code below is needed to create a sPyNNaker configuration file, which is needed on the first run of the software. It will be necessary to run this each time you start the Lab (it can also be added as the ...). A future image may already ... to do this.

```
def check_should_create():
    """
    Detects if there is an existing cfg file and if so if it
    :return:
    """
    if not os.path.isfile(FULL_PATH):
        return True
    if OVERRIDE_EXISTING:
        return True
```

Open Setup Notebook - Run

The screenshot displays the JupyterLab environment. The left sidebar shows a file browser with the path `/ SpiNNakerJupyterExamples / 00.Setup /` and a file named `Setup.ipynb` last modified 2 minutes ago. The main area shows the notebook `Setup.ipynb` with the following content:

```
# Make sure there is a spynaker.cfg file
if check_should_create():
    with open(FULL_PATH, 'w') as cfg:
        cfg.write("[Machine]\n")
        cfg.write("spalloc_server = https://spinnaker.cs.man.\n")
        cfg.write("\n")
        cfg.write("[Java]\n")
        cfg.write("use_java=True\n")
        cfg.write("\n")
        cfg.write("[Reports]\n")
        cfg.write("read_provenance_data = False\n")
        print(f"New {FULL_PATH} created")
```

Below the code cell, a text cell contains the following message, which is circled in red:

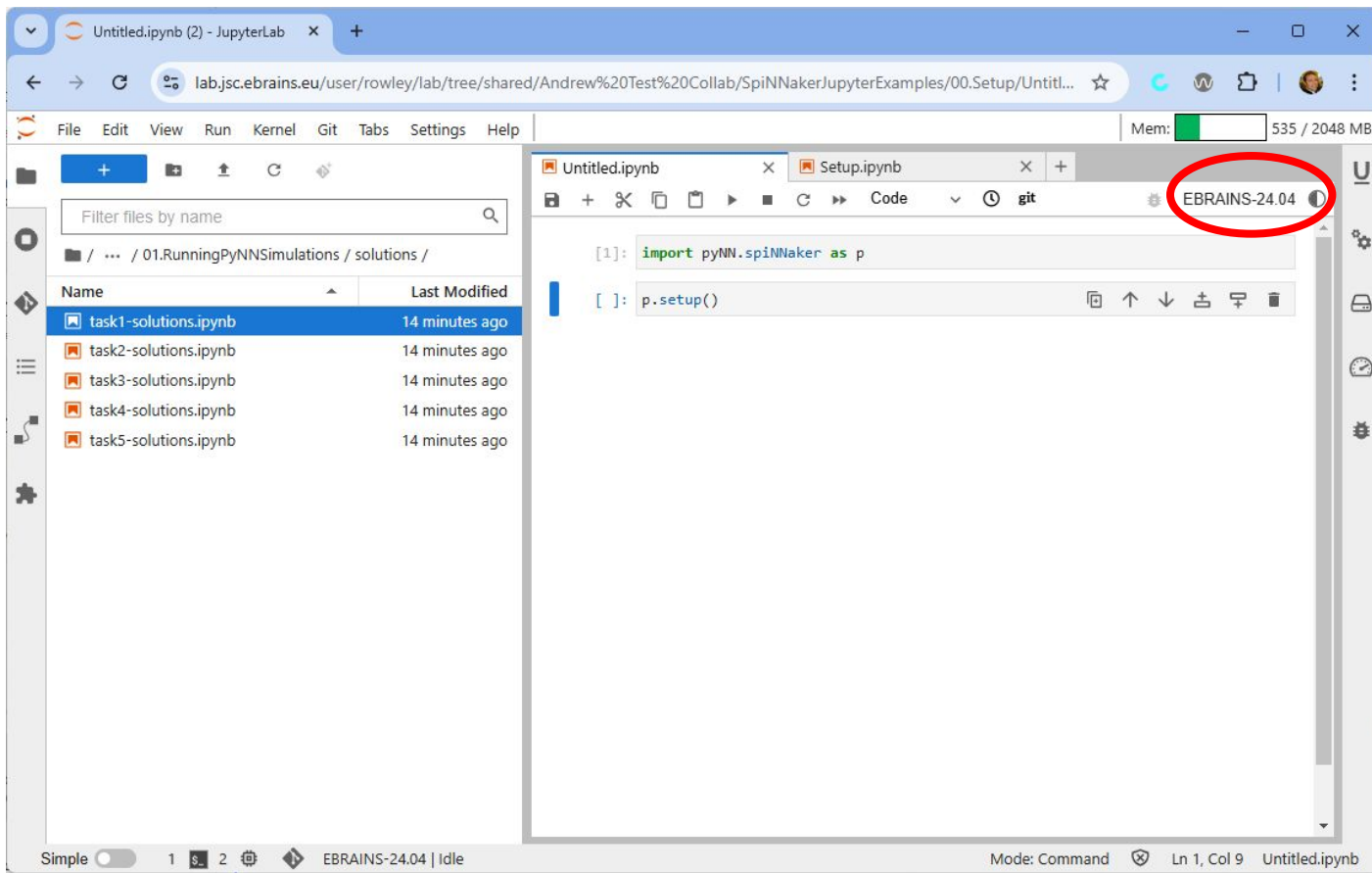
`/opt/app-root/src/.spynaker.cfg` already exists.
To replace it change `OVERRIDE_EXISTING` to `True`

The right sidebar shows a table of contents with the following items:

1. Running PyNN Simulations on SpiNNaker
2. Live Input and Output

The bottom status bar indicates the current mode is Command, the file is `Setup.ipynb`, and the memory usage is 315.93 / 2048.00 MB.

Changing Kernel



The screenshot shows a JupyterLab interface with the following components:

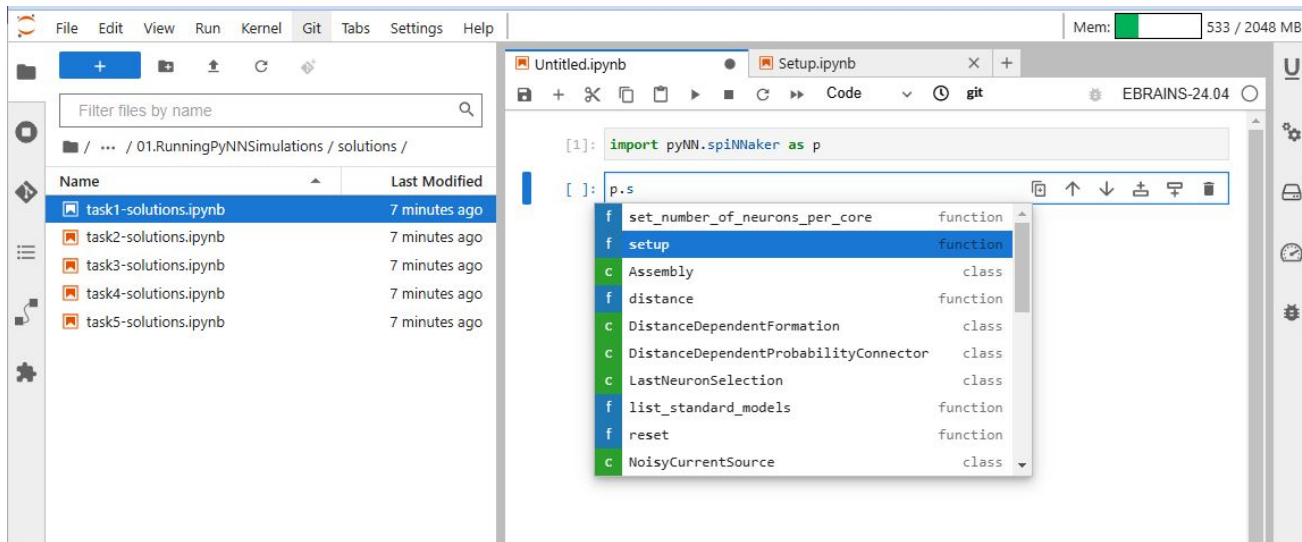
- Browser Address Bar:** `lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterExamples/00.Setup/Untitl...`
- File Explorer (Left):**
 - Path: `/ 01.RunningPyNNSimulations / solutions /`
 - Files: `task1-solutions.ipynb`, `task2-solutions.ipynb`, `task3-solutions.ipynb`, `task4-solutions.ipynb`, `task5-solutions.ipynb` (all modified 14 minutes ago).
- Kernel Menu (Top):** Opened, showing the current kernel as `EBRAINS-24.04`, which is circled in red.
- Code Editor (Right):**
 - Tab: `Setup.ipynb`
 - Code:

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

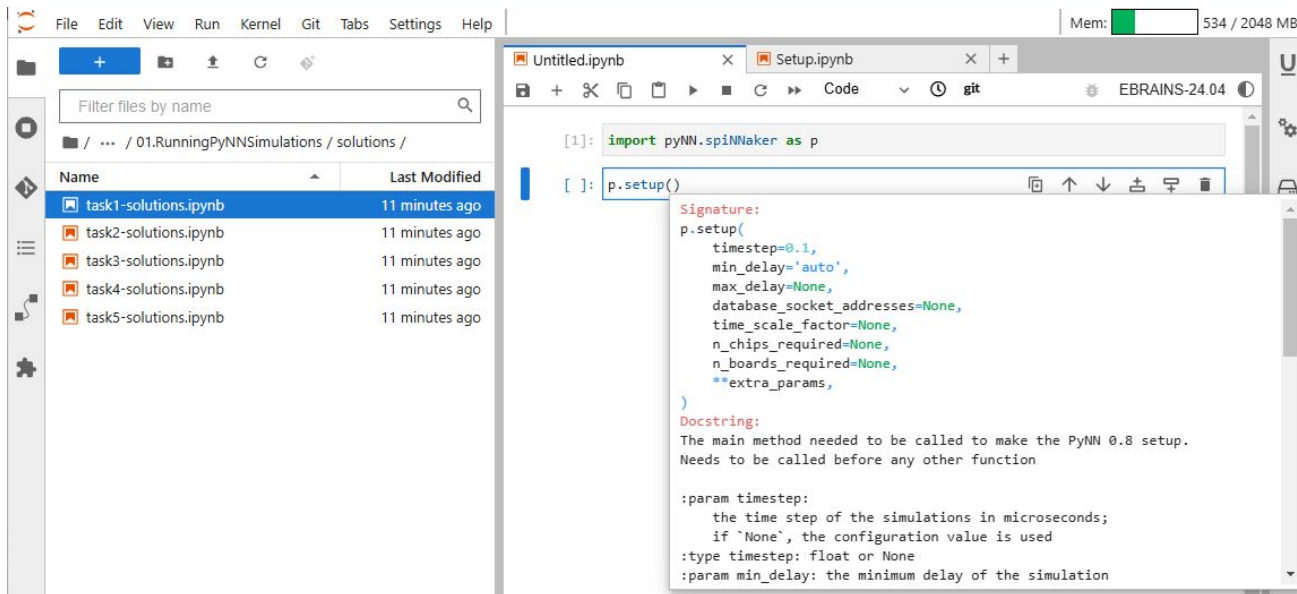


```
[ ]: p.setup()
```
- Bottom Status Bar:**
 - Left: `Simple` (toggle), `1`, `2`, `EBRAINS-24.04 | Idle`
 - Right: `Mode: Command`, `Ln 1, Col 9`, `Untitled.ipynb`

Jupyter Autocomplete



Jupyter Function Help



The screenshot shows the JupyterLab interface. On the left is a file browser showing a directory structure with files like 'task1-solutions.ipynb' through 'task5-solutions.ipynb'. The main area is a code editor with two open notebooks: 'Untitled.ipynb' and 'Setup.ipynb'. The 'Setup.ipynb' notebook is active, showing two code cells. The first cell contains `import pyNN.spiNNaker as p`. The second cell contains `p.setup()`. A tooltip is displayed over the `p.setup()` call, showing the function signature and docstring.

Signature:

```
p.setup(
    timestep=0.1,
    min_delay='auto',
    max_delay=None,
    database_socket_addresses=None,
    time_scale_factor=None,
    n_chips_required=None,
    n_boards_required=None,
    **extra_params,
)
```

Docstring:

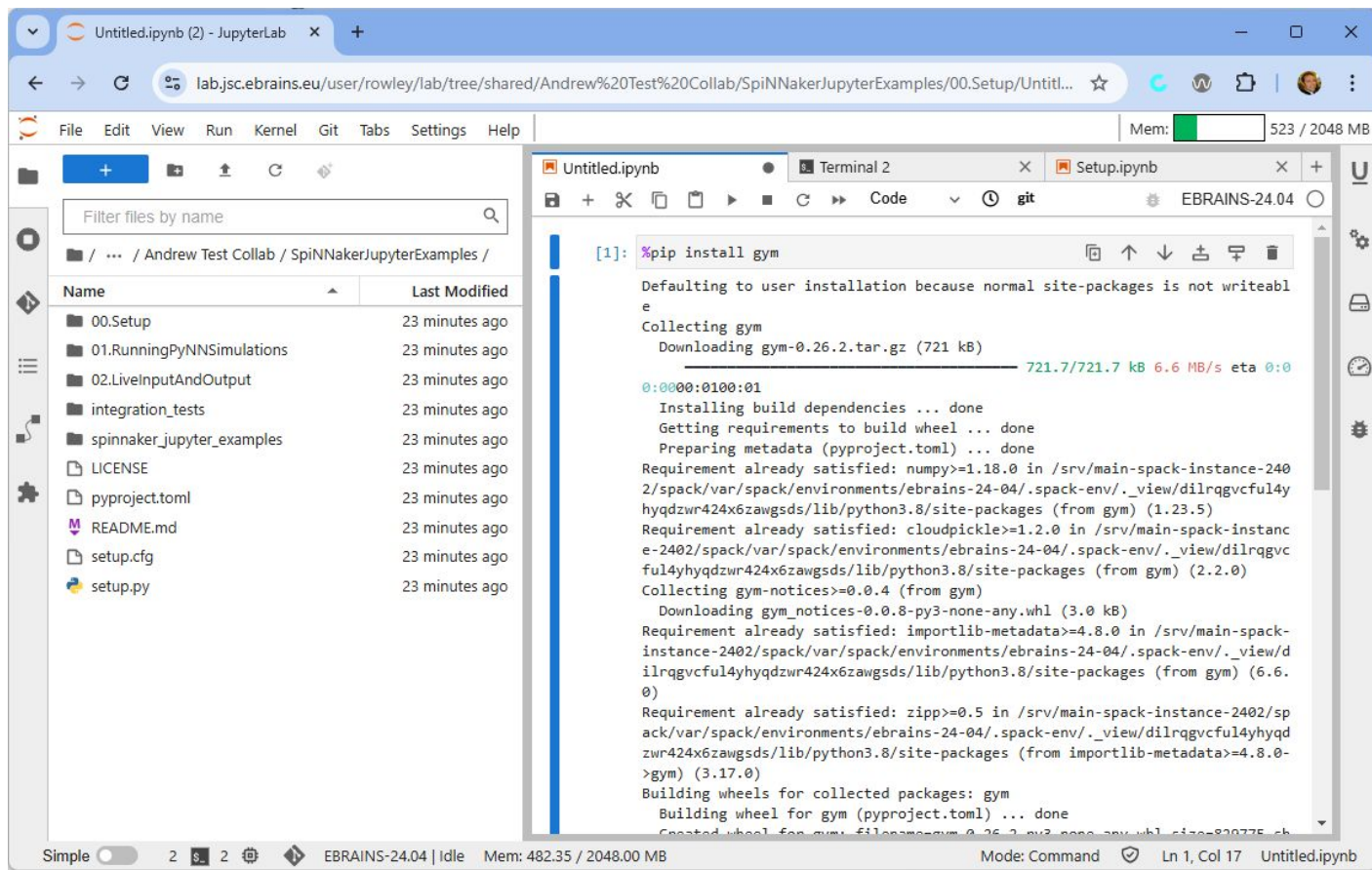
The main method needed to be called to make the PyNN 0.8 setup.
Needs to be called before any other function

Parameters:

- `timestep`: the time step of the simulations in microseconds; if 'None', the configuration value is used
- `timestep`: type float or None
- `min_delay`: the minimum delay of the simulation



Install Libraries



The screenshot shows a JupyterLab environment with a file browser on the left and a terminal window on the right. The file browser displays a directory structure for 'Andrew Test Collab / SpiNNakerJupyterExamples /'. The terminal window is running the command `pip install gym` and shows the progress of downloading and installing the gym library.

File Browser:

Name	Last Modified
00.Setup	23 minutes ago
01.RunningPyNNSimulations	23 minutes ago
02.LiveInputAndOutput	23 minutes ago
integration_tests	23 minutes ago
spinnaker_jupyter_examples	23 minutes ago
LICENSE	23 minutes ago
pyproject.toml	23 minutes ago
README.md	23 minutes ago
setup.cfg	23 minutes ago
setup.py	23 minutes ago

Terminal Window:

```
[1]: %pip install gym

Defaulting to user installation because normal site-packages is not writeable
Collecting gym
  Downloading gym-0.26.2.tar.gz (721 kB)
    721.7/721.7 kB 6.6 MB/s eta 0:00
0:0000:0100:01
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: numpy>=1.18.0 in /srv/main-spack-instance-2402/spack/var/spack/environments/ebbrains-24-04/.spack-env/_view/dilrqgvcful4hyqdzwr424x6zawgsds/lib/python3.8/site-packages (from gym) (1.23.5)
Requirement already satisfied: cloudpickle>=1.2.0 in /srv/main-spack-instance-2402/spack/var/spack/environments/ebbrains-24-04/.spack-env/_view/dilrqgvcful4hyqdzwr424x6zawgsds/lib/python3.8/site-packages (from gym) (2.2.0)
Collecting gym-notices>=0.0.4 (from gym)
  Downloading gym_notices-0.0.8-py3-none-any.whl (3.0 kB)
Requirement already satisfied: importlib-metadata>=4.8.0 in /srv/main-spack-instance-2402/spack/var/spack/environments/ebbrains-24-04/.spack-env/_view/dilrqgvcful4hyqdzwr424x6zawgsds/lib/python3.8/site-packages (from gym) (6.6.0)
Requirement already satisfied: zipp>=0.5 in /srv/main-spack-instance-2402/spack/var/spack/environments/ebbrains-24-04/.spack-env/_view/dilrqgvcful4hyqdzwr424x6zawgsds/lib/python3.8/site-packages (from importlib-metadata>=4.8.0->gym) (3.17.0)
Building wheels for collected packages: gym
  Building wheel for gym (pyproject.toml) ... done
  Created wheel for gym: filename=gym-0.26.2-py3-none-any.whl size=820775 sha256=...
```

Bottom Bar:

Simple 2 2 EBRAINS-24.04 | Idle Mem: 482.35 / 2048.00 MB Mode: Command Ln 1, Col 17 Untitled.ipynb

Terminal Access

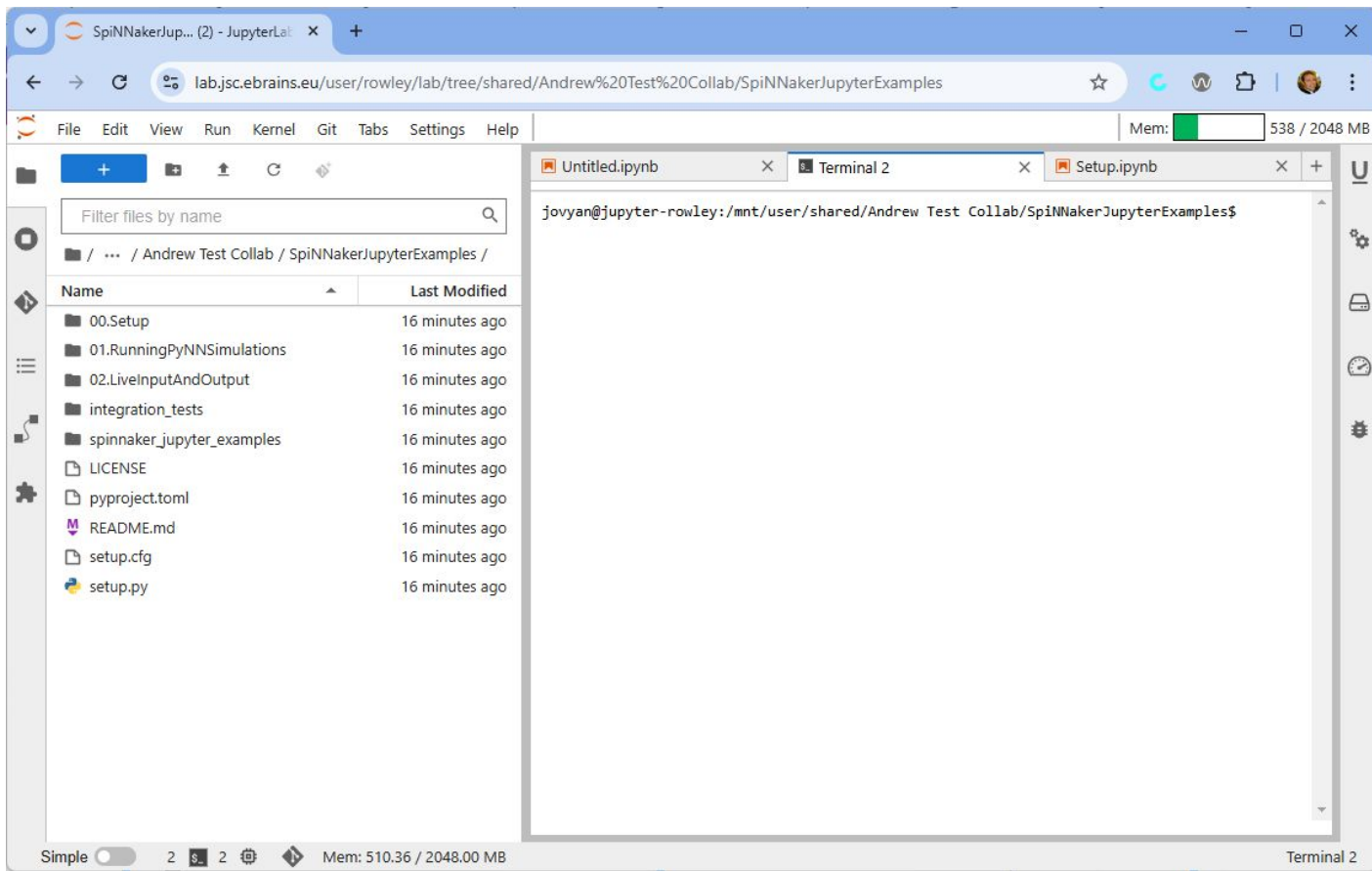
The screenshot displays the JupyterLab web interface in a browser. The address bar shows the URL: `lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterExamples`. The interface includes a top menu bar with options like File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. A memory usage indicator shows 535 / 2048 MB.

On the left, a file browser pane shows the directory structure: `/ ... / Andrew Test Collab / SpiNNakerJupyterExamples /`. It lists files and folders such as `00.Setup`, `01.RunningPyNNSimulations`, `02.LiveInputAndOutput`, `integration_tests`, `spinnaker_jupyter_examples`, `LICENSE`, `pyproject.toml`, `README.md`, `setup.cfg`, and `setup.py`, all marked as "15 minutes ago".

The main area is the Launcher, which displays various application icons. A red circle highlights the **Terminal** icon, which is represented by a black square with a white dollar sign and an underscore (`$ _`). Other visible icons include EBRAINS-experimental, R 3.6.3, R-EBRAINS-23.02, R-EBRAINS-23.06, R-EBRAINS-23.09, R-EBRAINS-24.04, R-EBRAINS-experimental, Text File, Xircuits File, Markdown File, Python File, R File, and Show Contextual Help.

The bottom status bar shows "Simple" mode, a tab indicator with "1" and "2", and a memory usage indicator showing "Mem: 509.67 / 2048.00 MB". The word "Launcher" is visible in the bottom right corner.

Terminal Access



The screenshot displays a JupyterLab environment. The top bar shows the browser address: `lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterExamples`. The left sidebar contains a file browser with a search bar and a list of files and folders. The right pane shows a terminal window with the prompt `jovyan@jupyter-rowley:/mnt/user/shared/Andrew Test Collab/SpiNNakerJupyterExamples$`.

File Browser Contents:

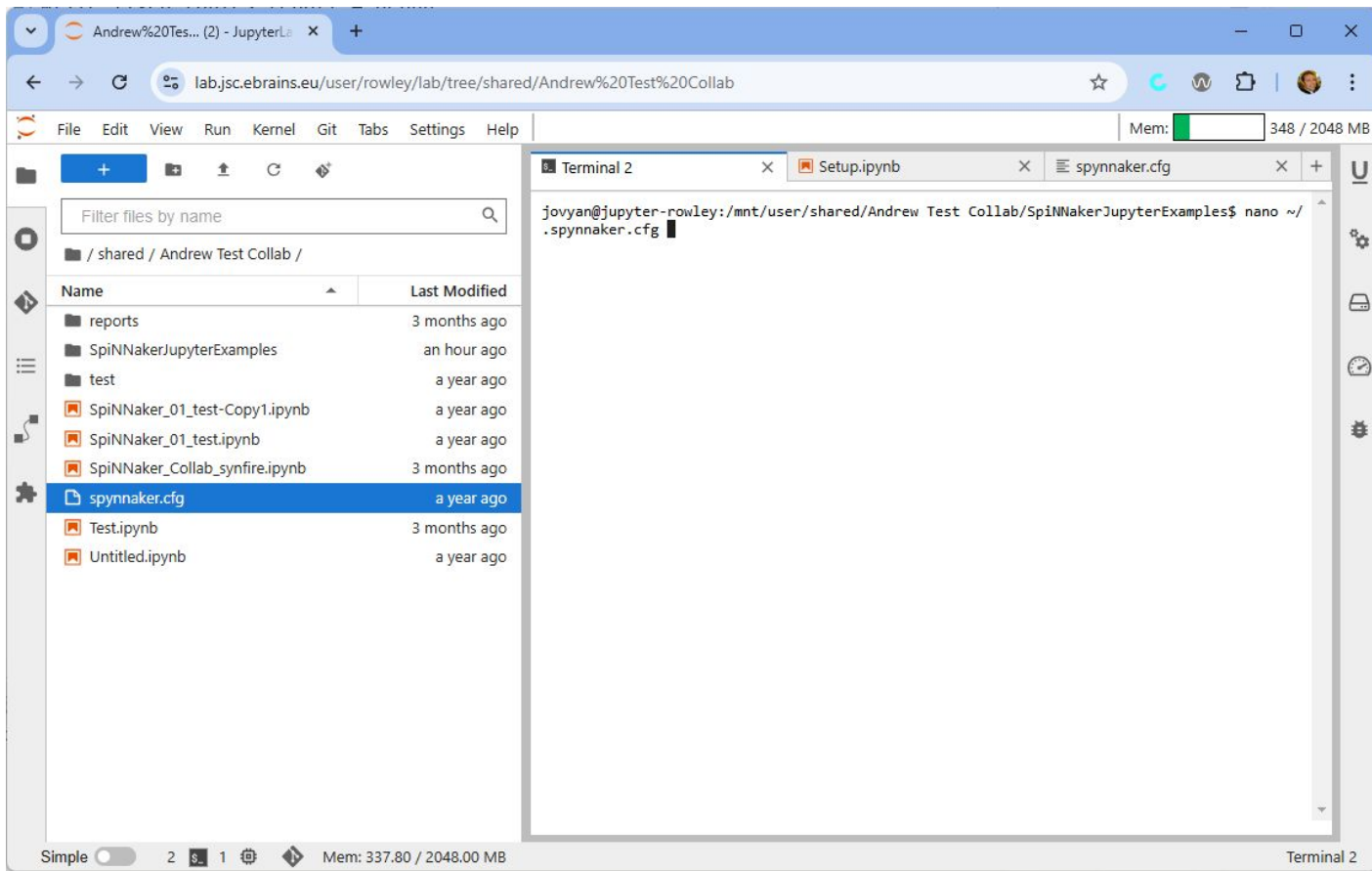
Name	Last Modified
00.Setup	16 minutes ago
01.RunningPyNNSimulations	16 minutes ago
02.LiveInputAndOutput	16 minutes ago
integration_tests	16 minutes ago
spinnaker_jupyter_examples	16 minutes ago
LICENSE	16 minutes ago
pyproject.toml	16 minutes ago
README.md	16 minutes ago
setup.cfg	16 minutes ago
setup.py	16 minutes ago

Terminal Window:

```
jovyan@jupyter-rowley:/mnt/user/shared/Andrew Test Collab/SpiNNakerJupyterExamples$
```

The bottom status bar indicates memory usage: `Mem: 510.36 / 2048.00 MB`.

Edit SpiNNaker Configuration



The screenshot displays a JupyterLab environment. The left sidebar shows a file browser with the following structure:

- Filter files by name
- / shared / Andrew Test Collab /
- reports (3 months ago)
- SpiNNakerJupyterExamples (an hour ago)
- test (a year ago)
- SpiNNaker_01_test-Copy1.ipynb (a year ago)
- SpiNNaker_01_test.ipynb (a year ago)
- SpiNNaker_Collab_synfire.ipynb (3 months ago)
- spynnaker.cfg** (a year ago)
- Test.ipynb (3 months ago)
- Untitled.ipynb (a year ago)

The right pane shows a terminal window with the following command prompt and command:

```
jovyan@jupyter-rowley:/mnt/user/shared/Andrew Test Collab/SpiNNakerJupyterExamples$ nano ~/spynnaker.cfg
```

The terminal window also shows tabs for 'Terminal 2', 'Setup.ipynb', and 'spynnaker.cfg'.

Set Reports Options

The screenshot displays a JupyterLab environment. On the left, a file browser shows a directory structure with 'drive' and 'shared' folders. The main area is a terminal window titled 'Terminal 2' showing the configuration of the 'spinnaker.cfg' file. The configuration includes settings for the machine, Java, and reports.

Terminal 2: GNU nano 4.8 /opt/app-root/src/.spynaker.cfg Modified

```
[Machine]
spalloc_server = https://spinnaker.cs.man.ac.uk/spalloc/

[Java]
use_java = True

[Reports]
default_report_file_path = /opt/app-root/src/
read_provenance_data = False
```

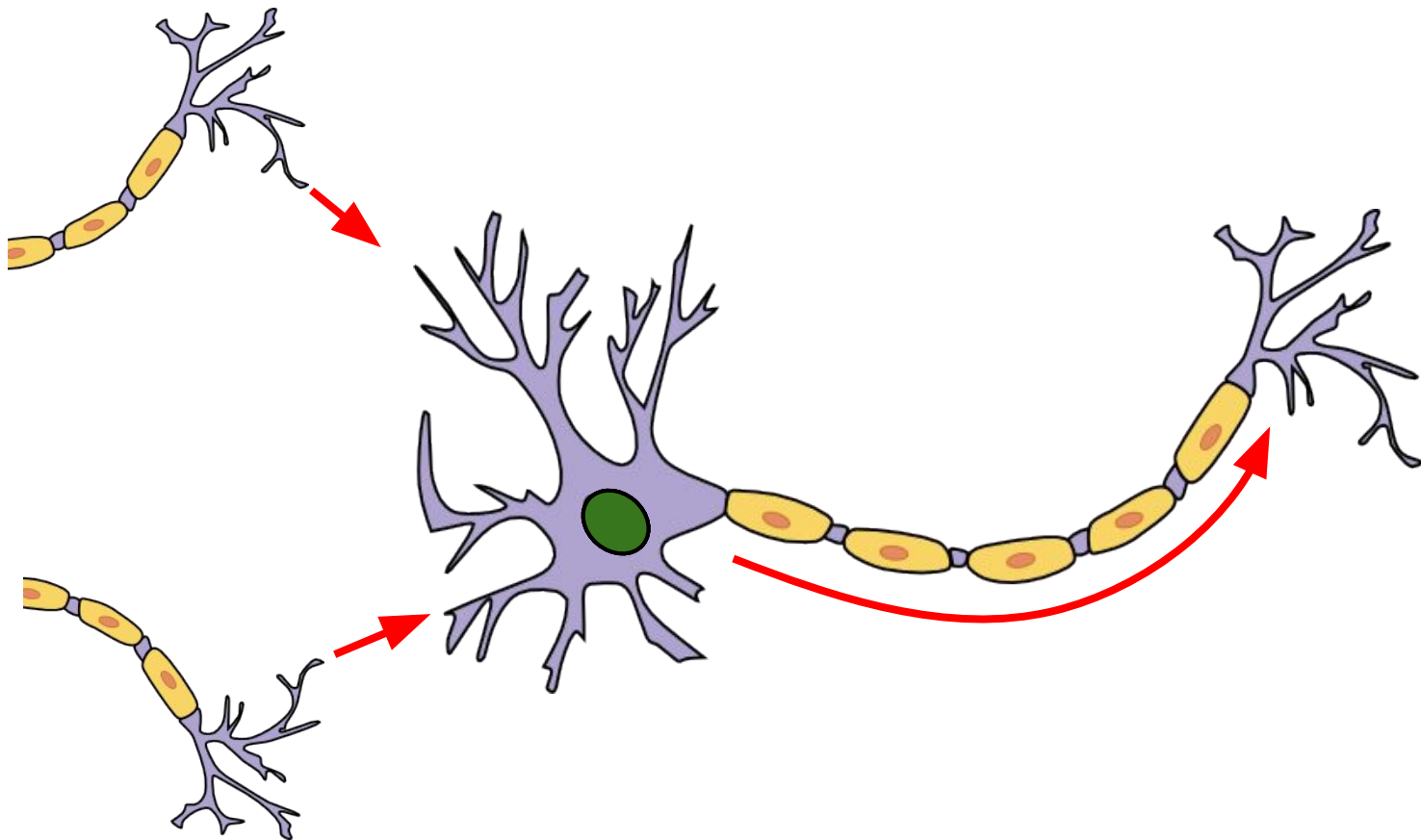
At the bottom of the terminal, a list of keyboard shortcuts is visible:

- Get Help**: `^G`
- Exit**: `^X`
- Write Out**: `^O`
- Read File**: `^R`
- Where Is**: `^W`
- Replace**: `^M`
- Cut Text**: `^K`
- Paste Text**: `^U`
- Cur Pos**: `^C`
- Go To Line**: `^G`
- Previous**: `^P`
- Next**: `^N`

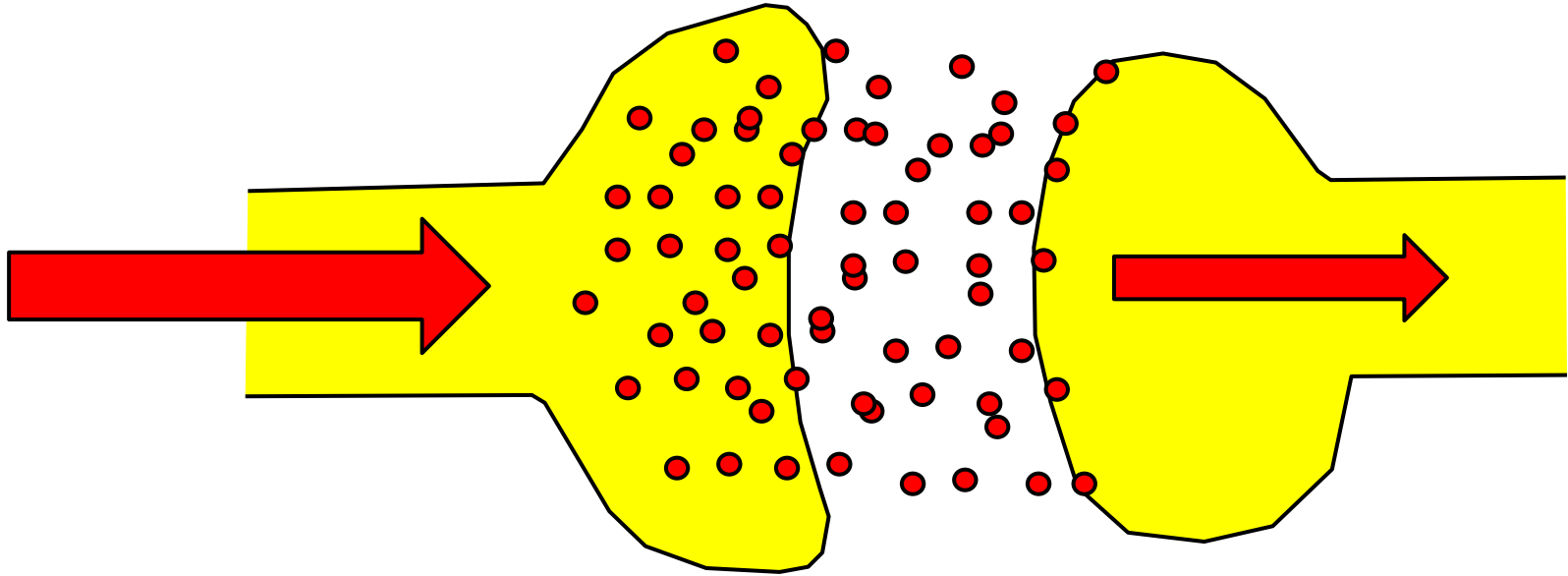
The bottom status bar shows the memory usage: 551.05 / 2048.00 MB.

Spiking Neural Networks

Spiking Neural Networks

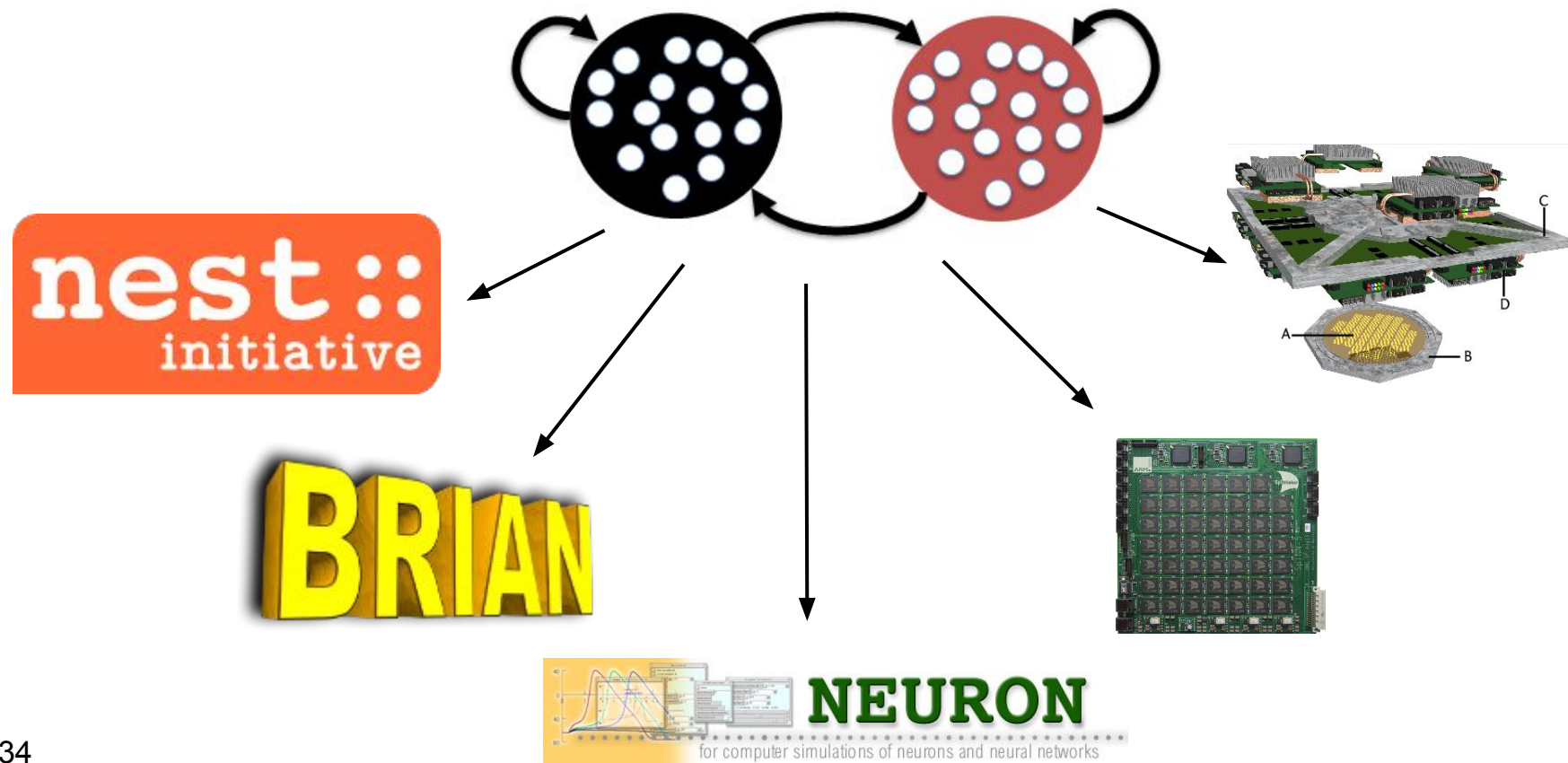


Spiking Neural Networks



PyNN

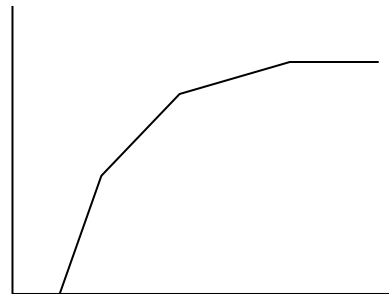
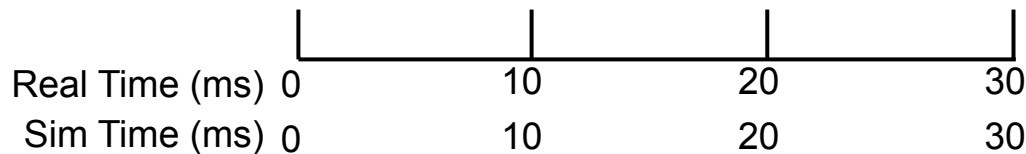
What is PyNN?



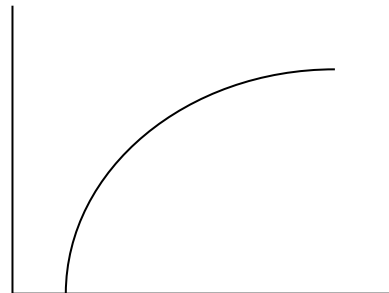
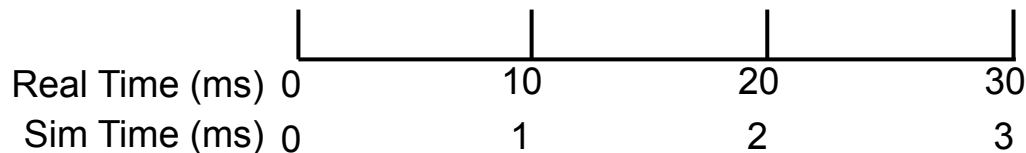
PyNN - Setup

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

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

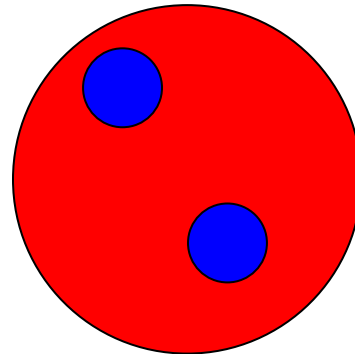
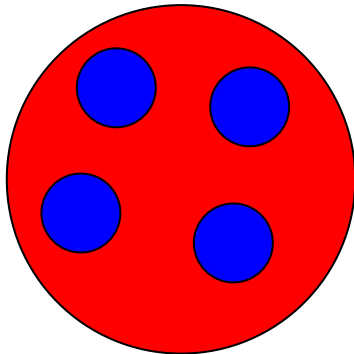


```
p.setup(timestep=0.1)
```



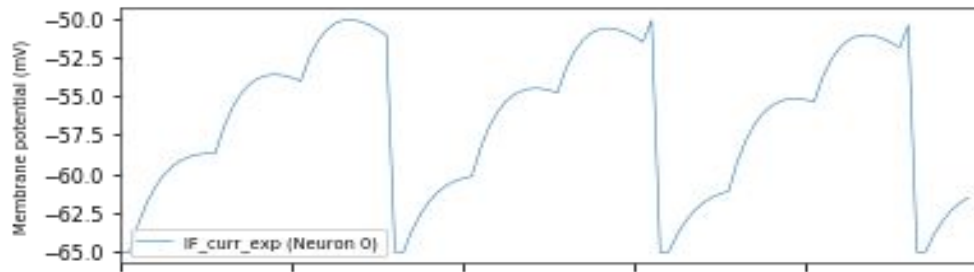
PyNN - Populations

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

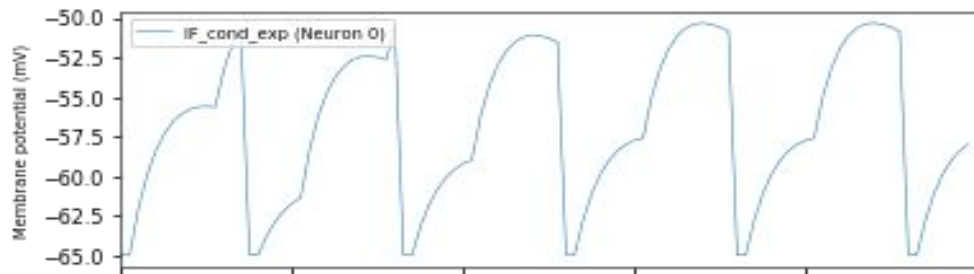


PyNN Populations - Models

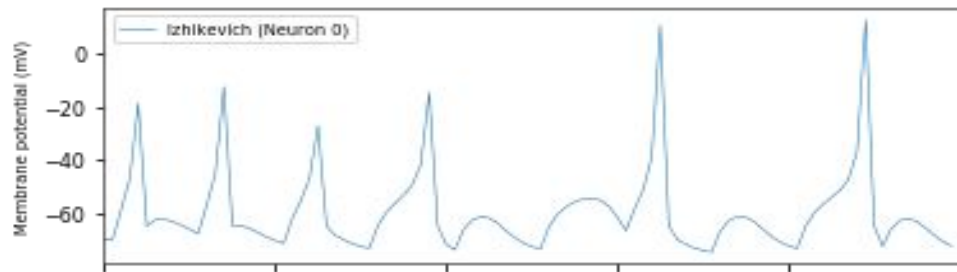
`p.IF_curr_exp(...)`



`p.IF_cond_exp(...)`

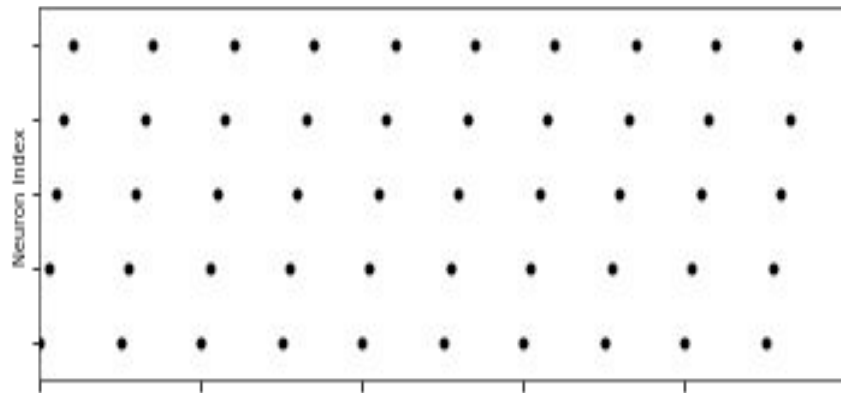


`p.Izhikevich(...)`

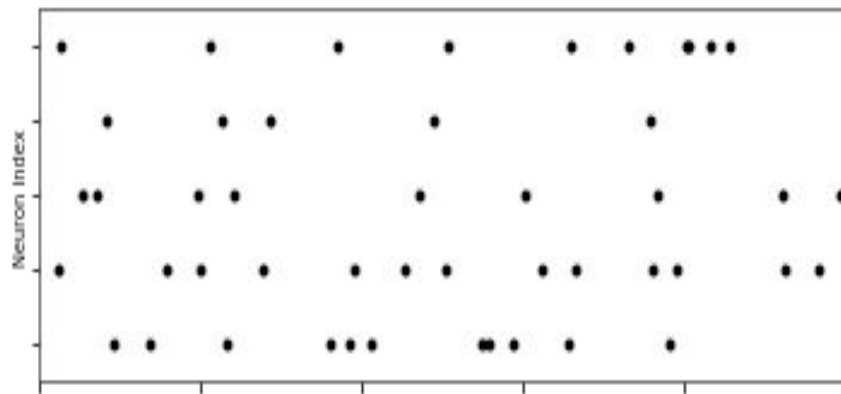


PyNN Populations - Models

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

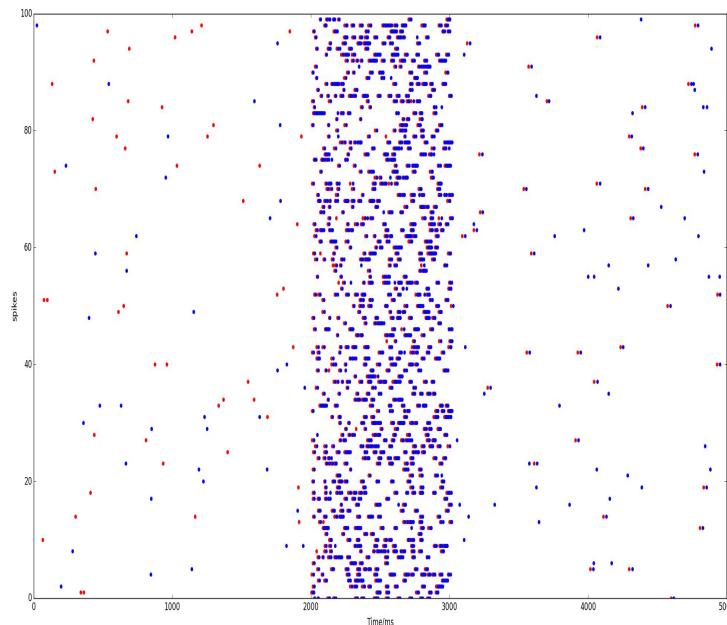
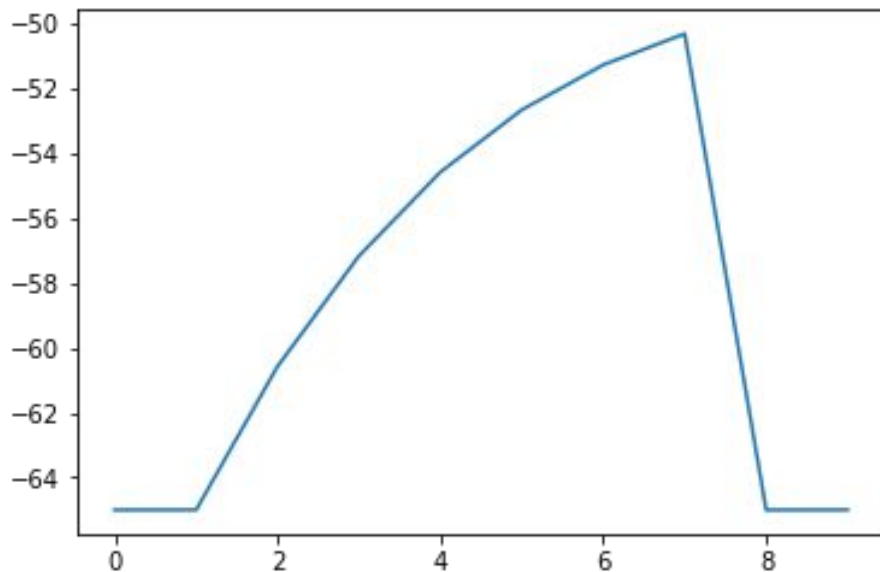


```
p.SpikeSourcePoisson(
    rate=...)
```



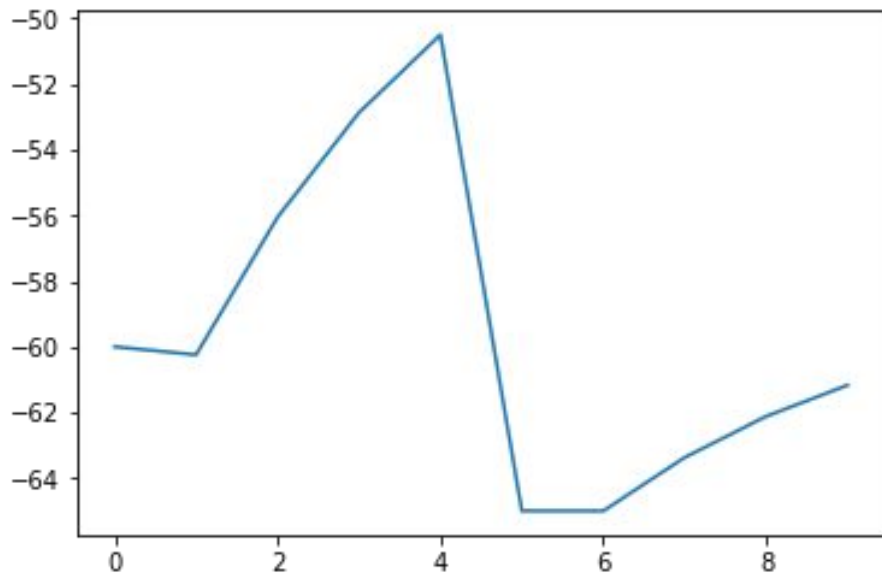
PyNN Populations - Recording

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



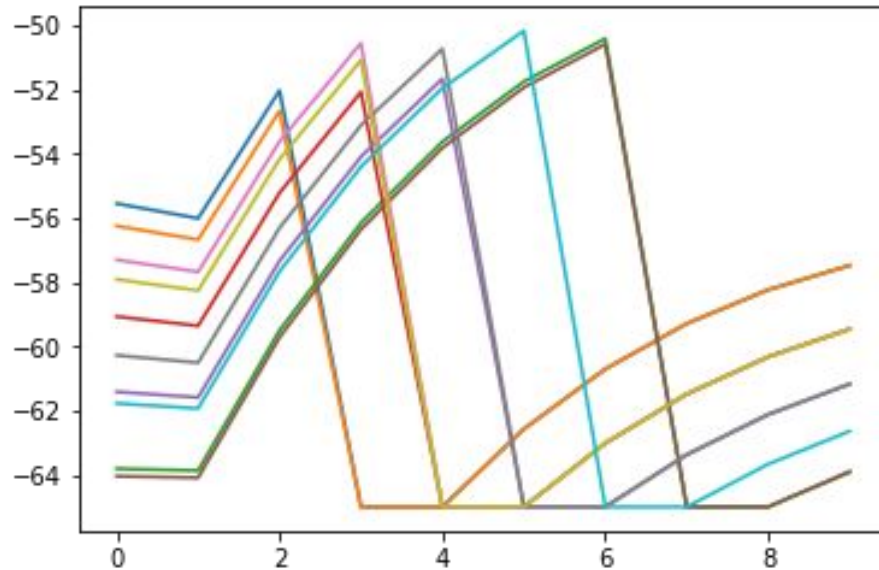
PyNN Populations - Initialize

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



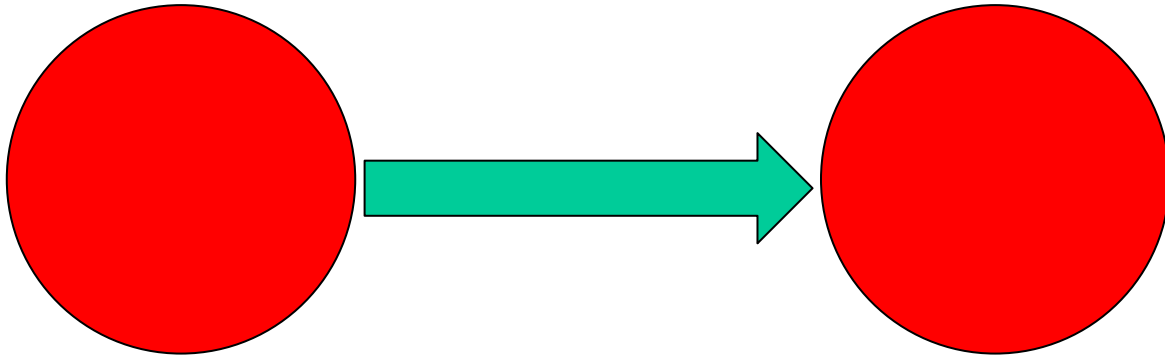
PyNN - Random Parameters

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



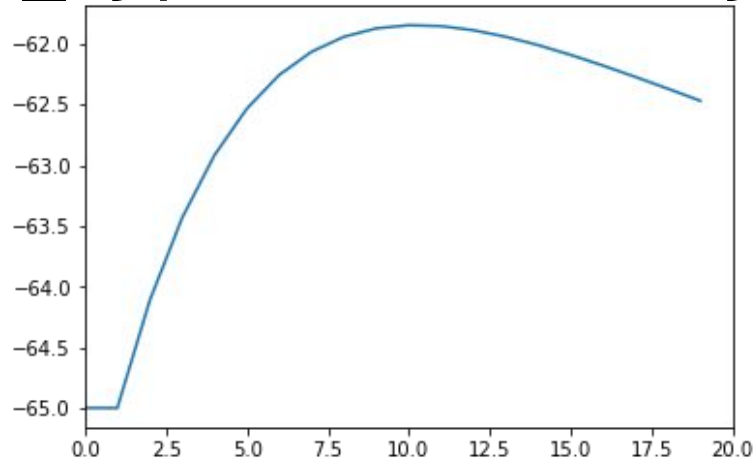
PyNN - Projections

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



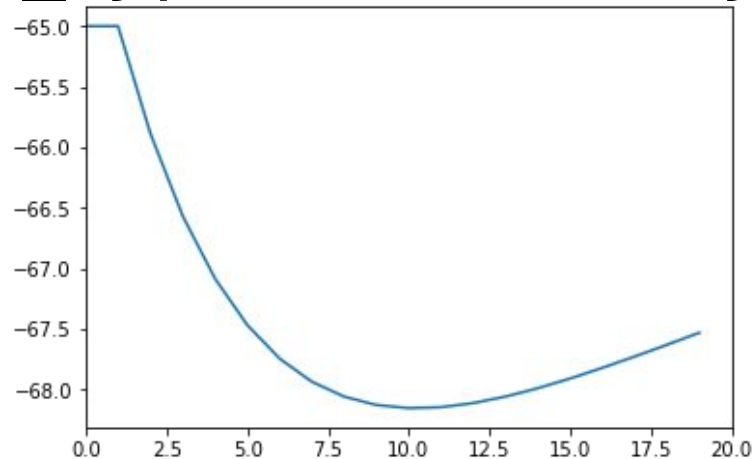
PyNN - Projections

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



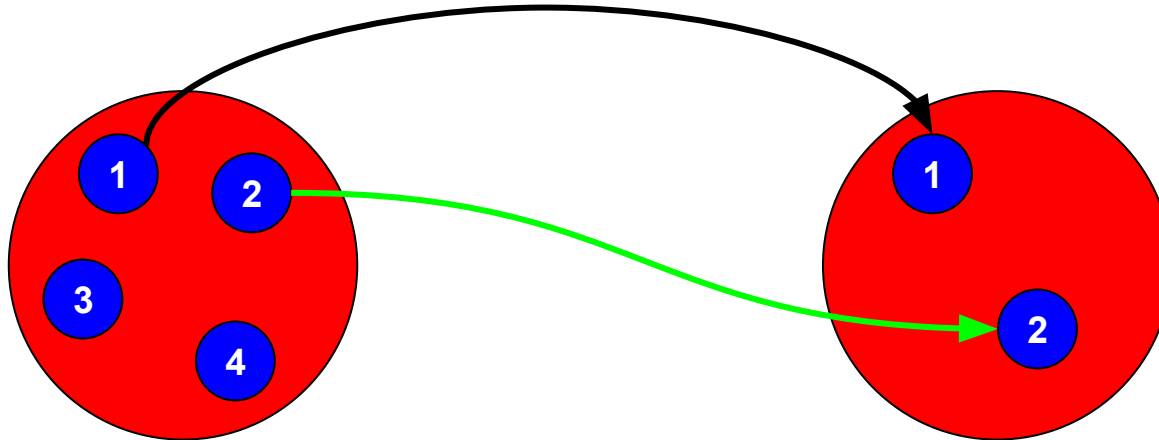
PyNN - Projections

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



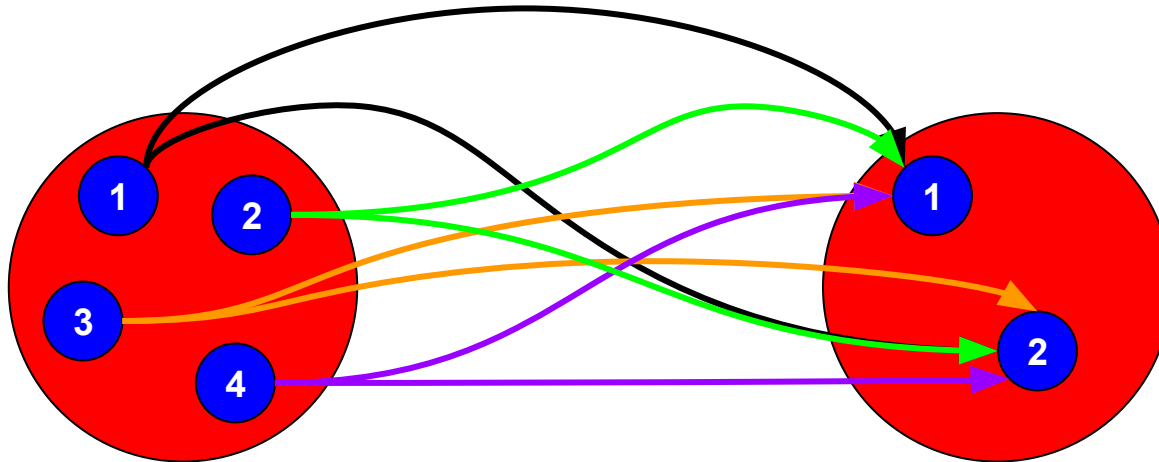
PyNN Projections - Connectors

```
p.OneToOneConnector()
```



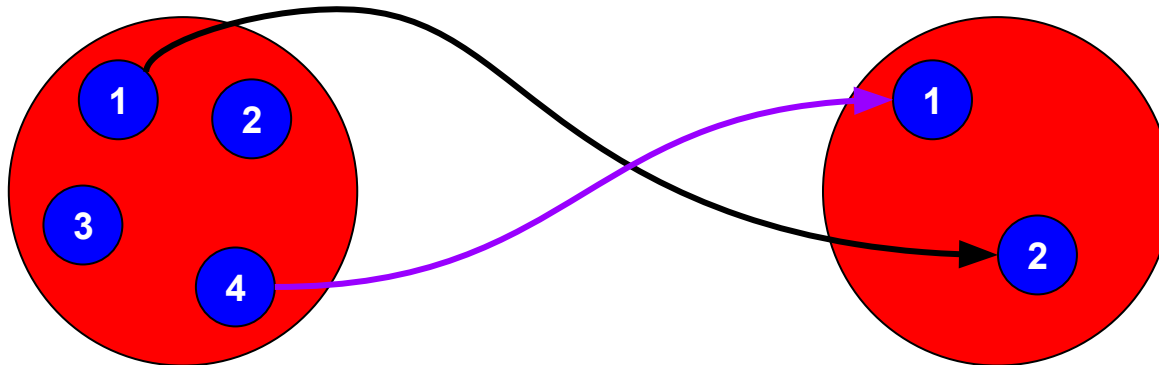
PyNN Projections - Connectors

```
p.AllToAllConnector()
```



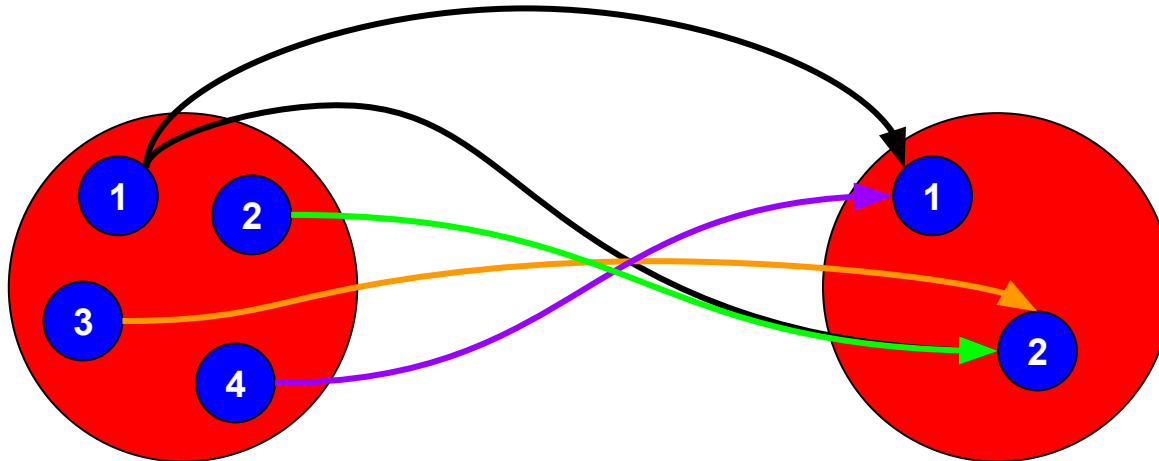
PyNN Projections - Connectors

`p.FixedProbabilityConnector(p=0.1)`



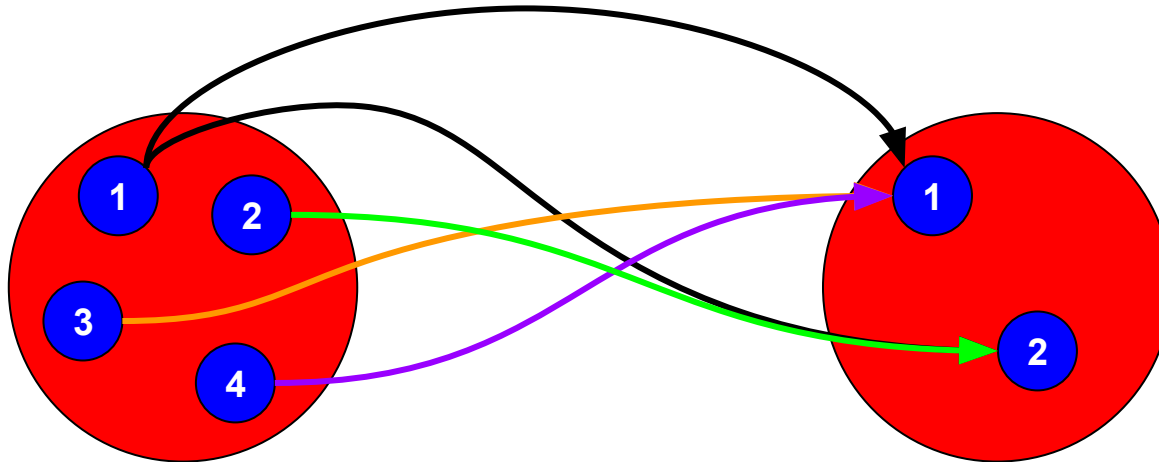
PyNN Projections - Connectors

`p.FixedProbabilityConnector(p=0.5)`



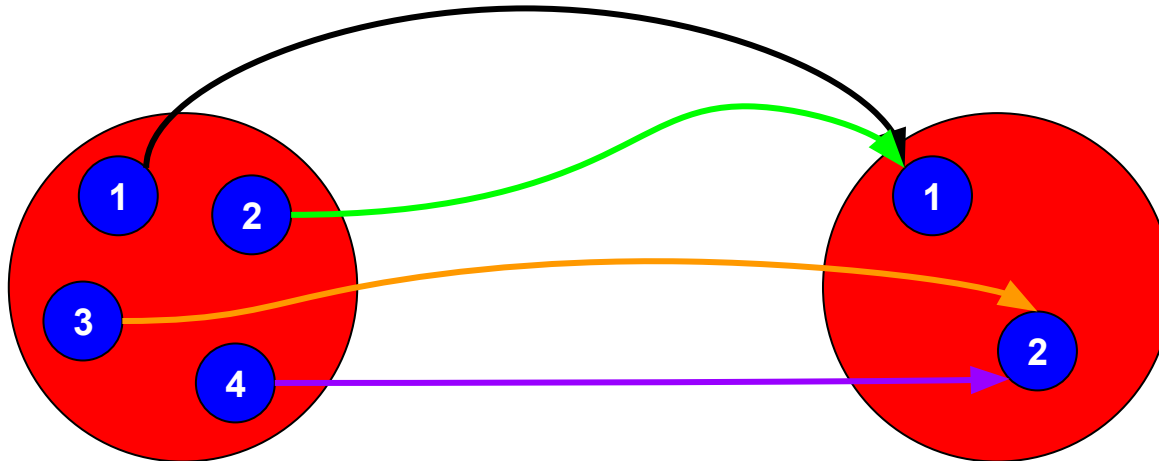
PyNN Projections - Connectors

```
p.FixedTotalNumberConnector(5)
```



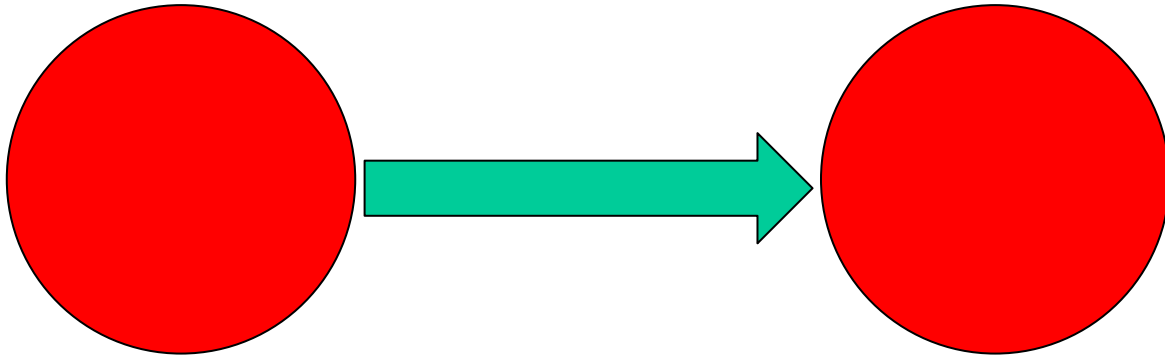
PyNN Projections - Connectors

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



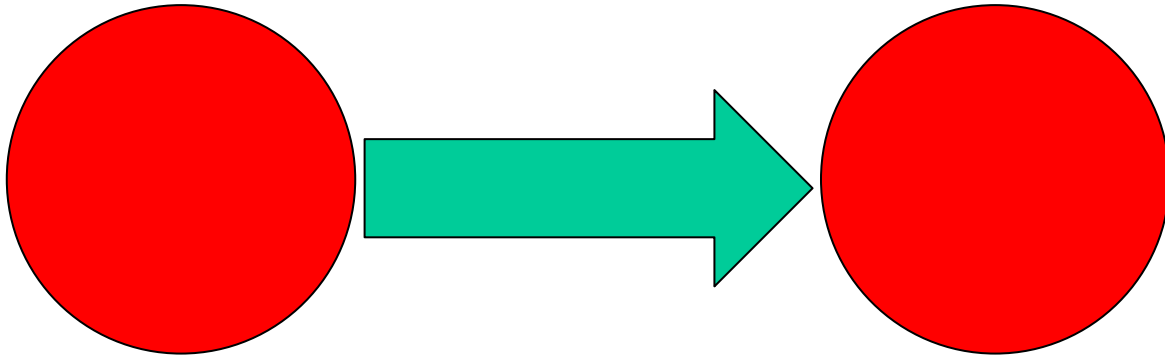
PyNN - Static Synapse Types

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



PyNN - Static Synapse Types

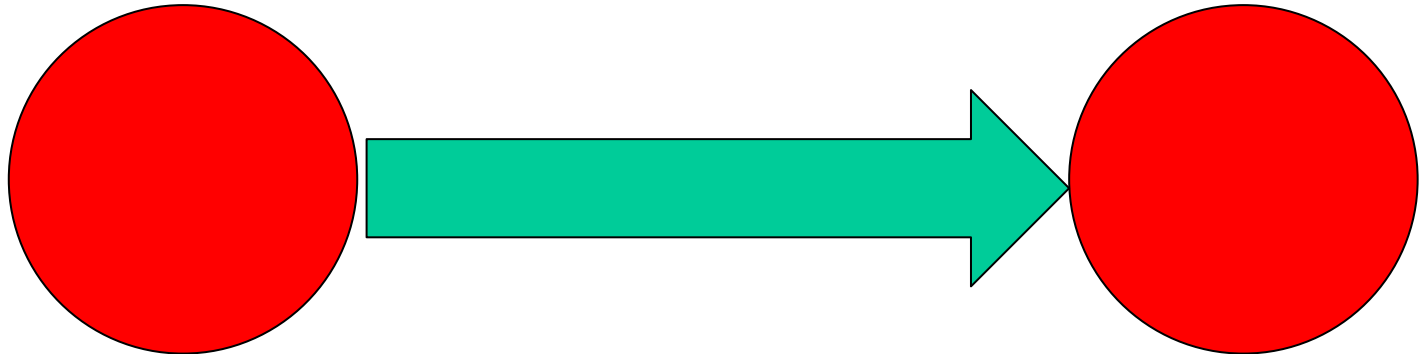
```
p.StaticSynapse(weight=5.0, delay=2.0)
```



PyNN - Static Synapse Types

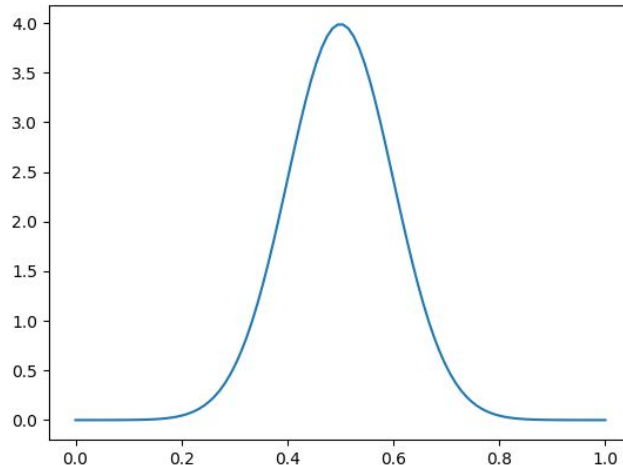
```
p.StaticSynapse(weight=5.0, delay=3.0)
```

(timestep <= delay <= 144 * timestep)



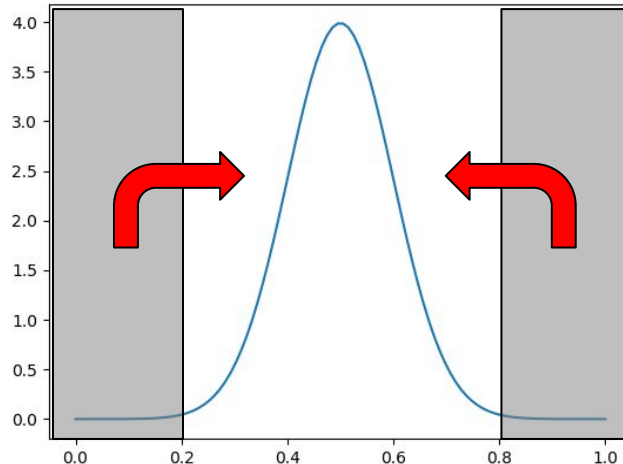
PyNN - Random Weights and Delays

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



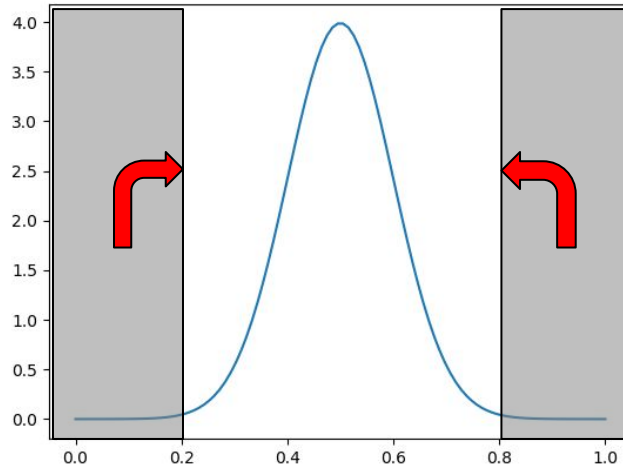
PyNN - Random Weights and Delays

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

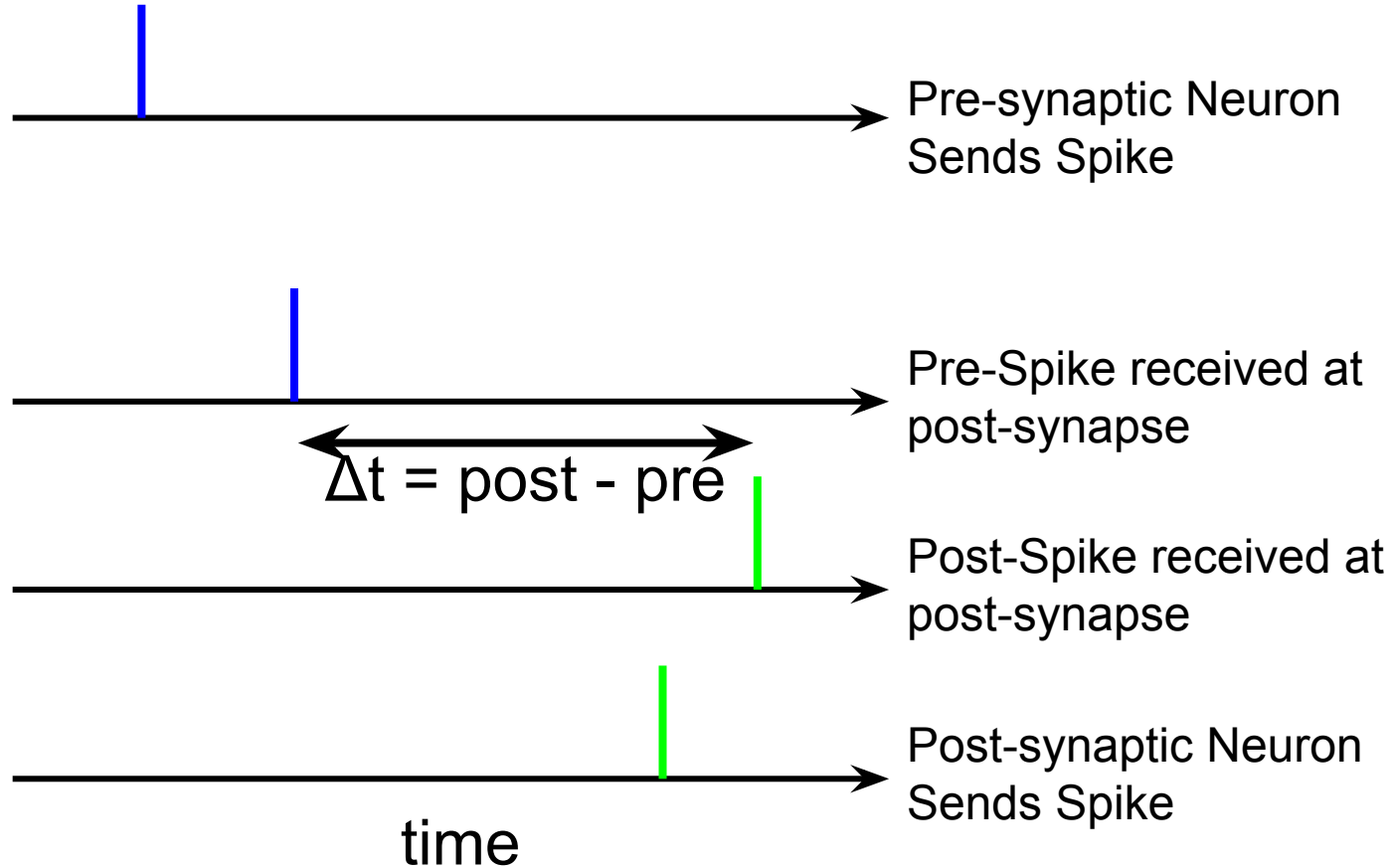
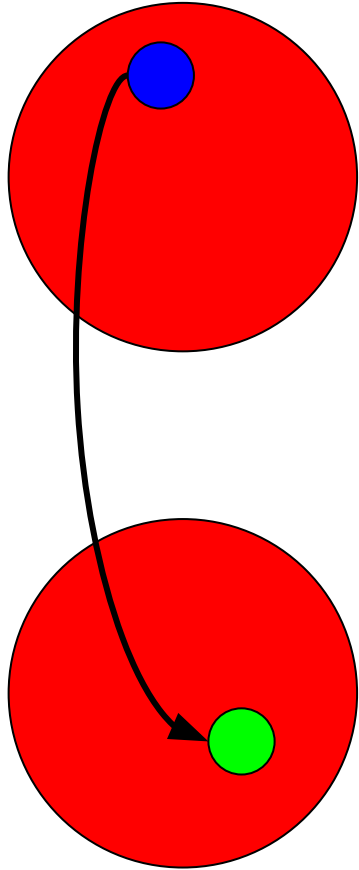


PyNN - Random Weights and Delays

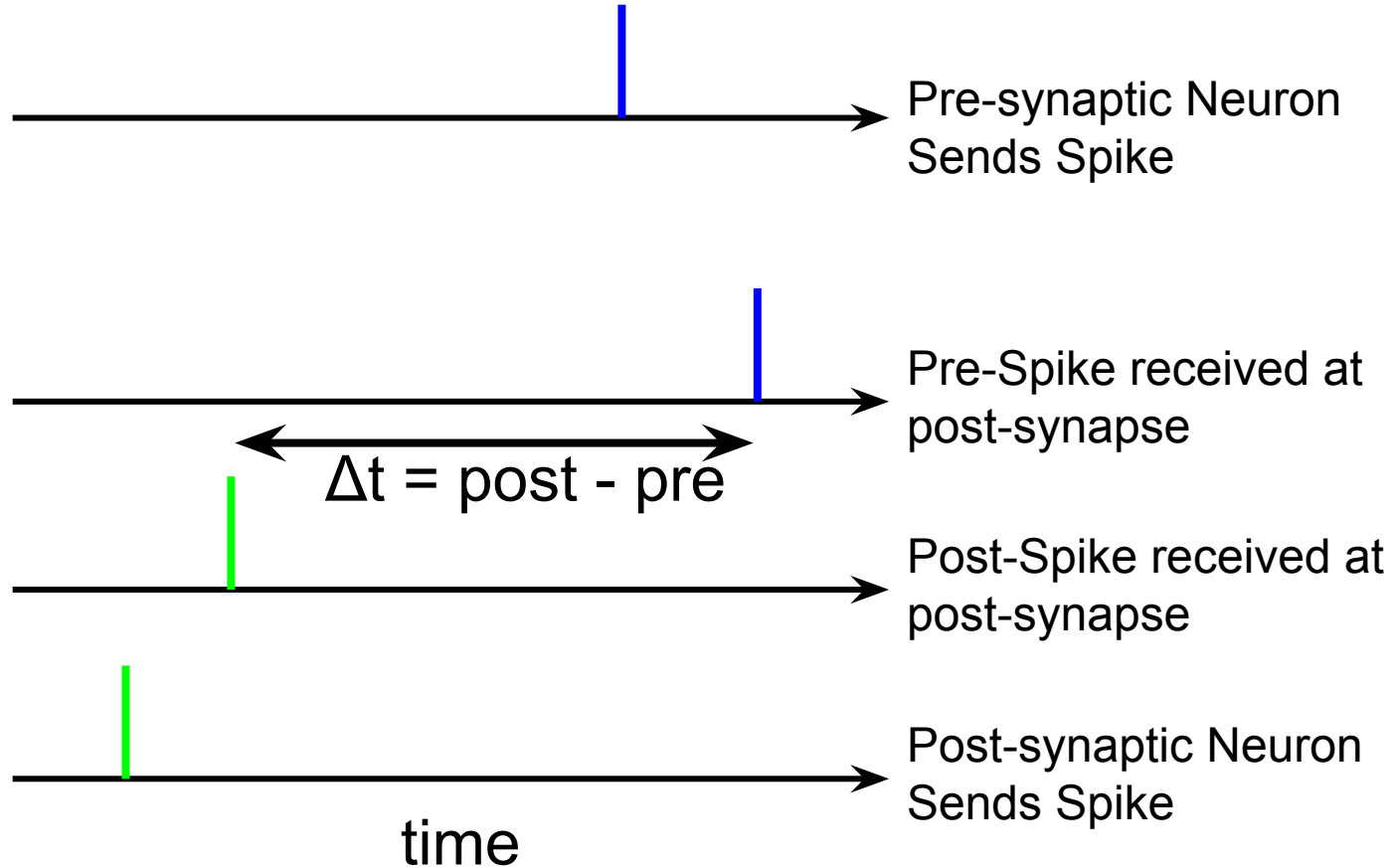
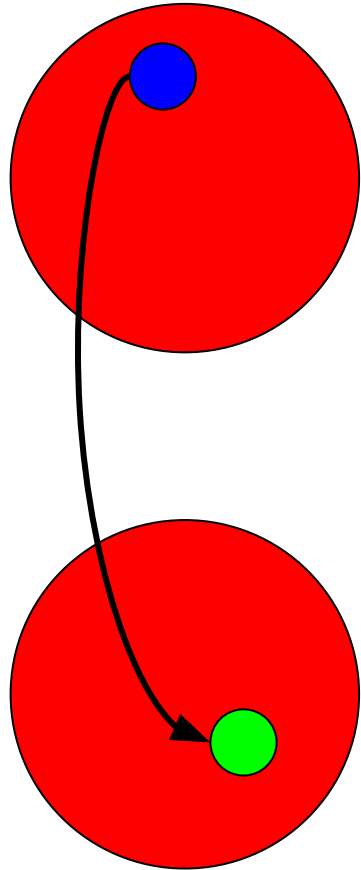
```
weight_dist = p.RandomDistribution(  
    "normal_clipped_to_boundary",  
    mu=0.5, sigma=0.1, low=0.2 high=0.8)
```



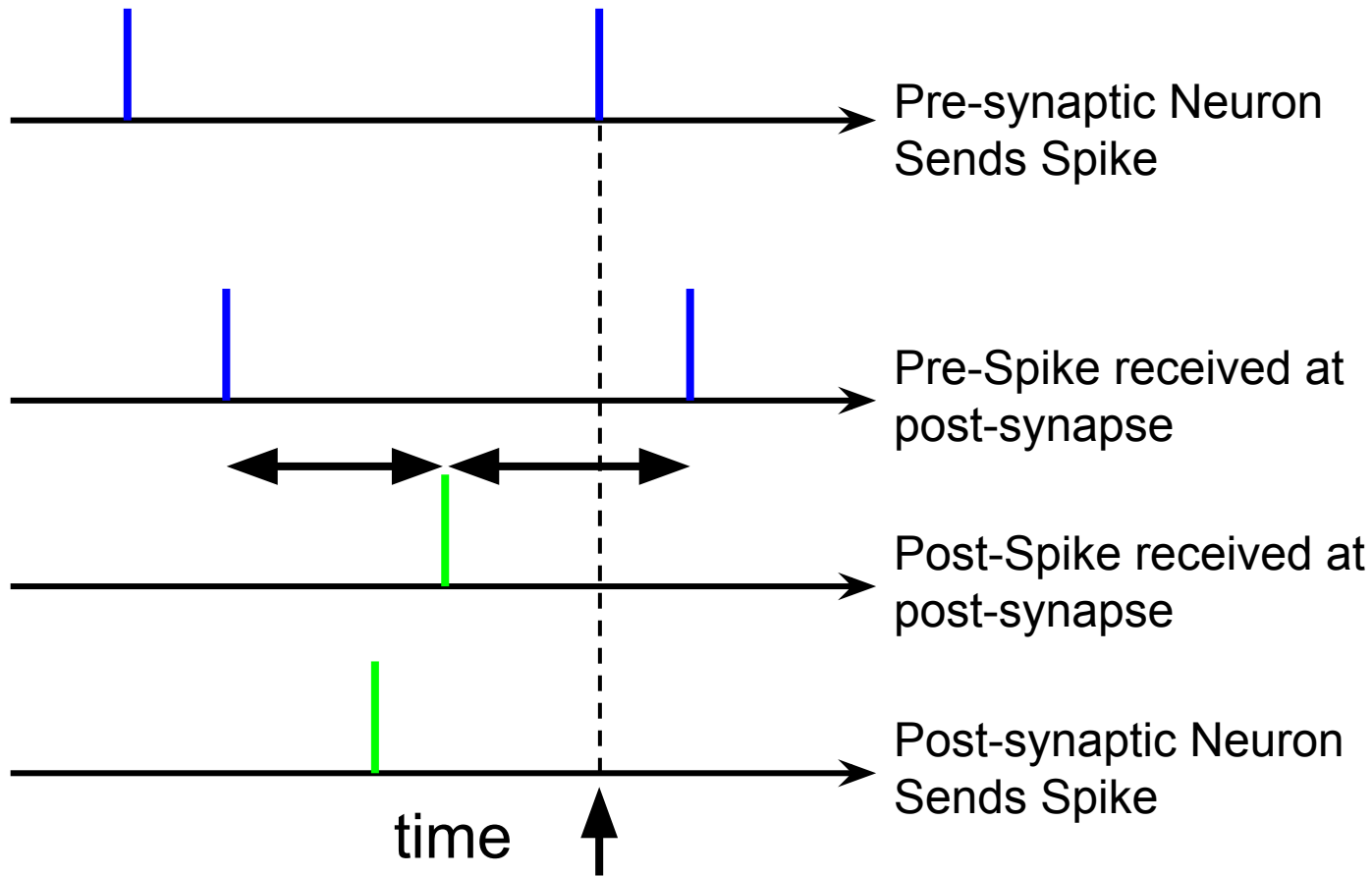
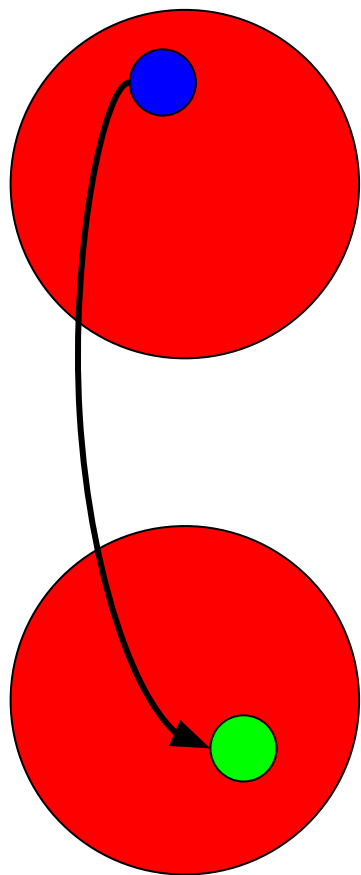
Spike Timing Dependent Plasticity



Spike Timing Dependent Plasticity

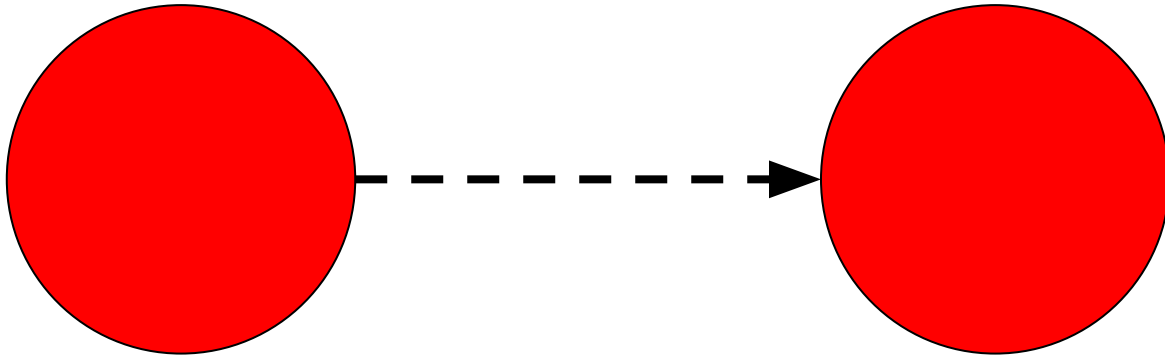


STDP - Deferred Execution



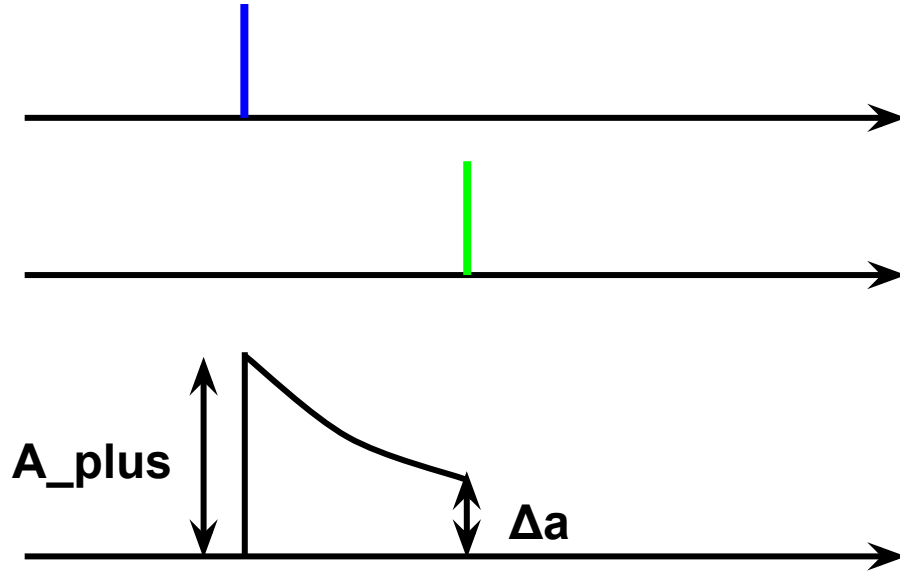
PyNN - STDP Synapse Types

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



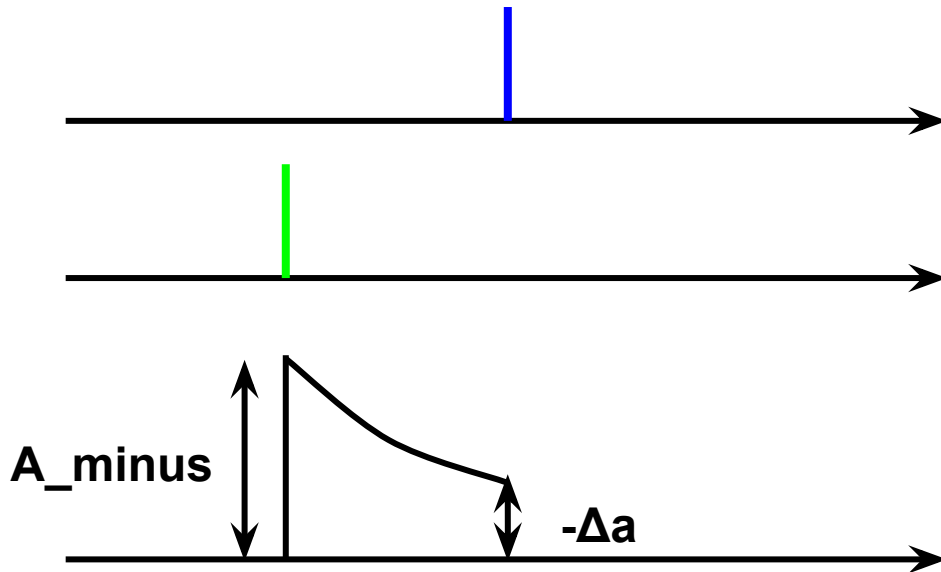
PyNN - Timing Dependence

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



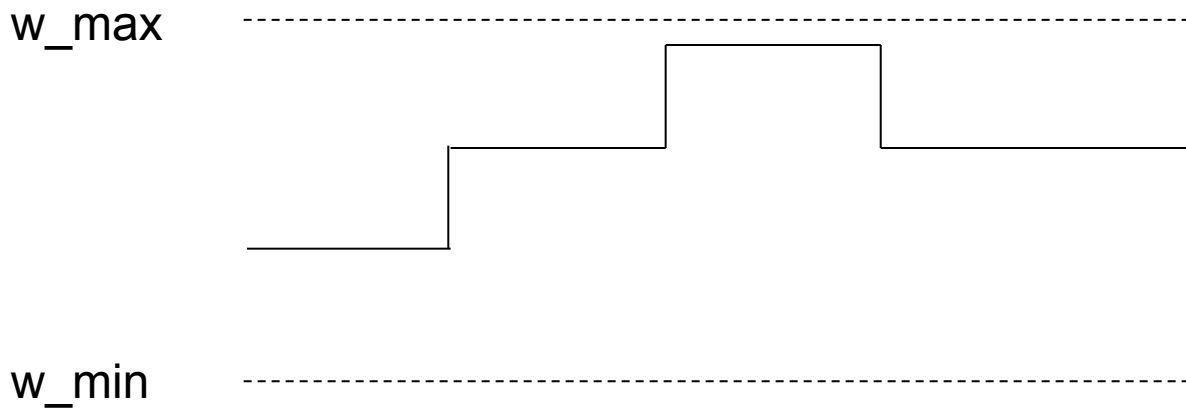
PyNN - Timing Dependence

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



PyNN - Weight Dependence

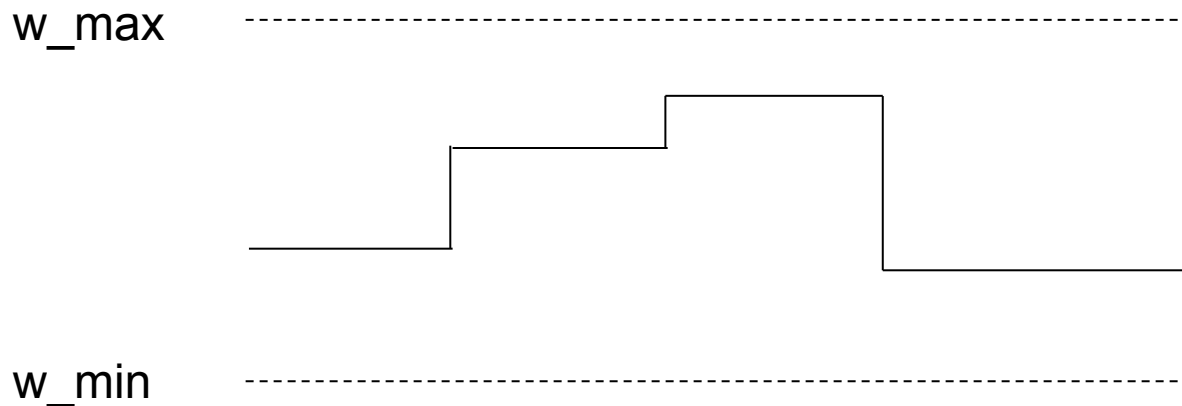
```
sim.AdditiveWeightDependence(w_max=5.0, w_min=0.0)
```



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

PyNN - Weight Dependence

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



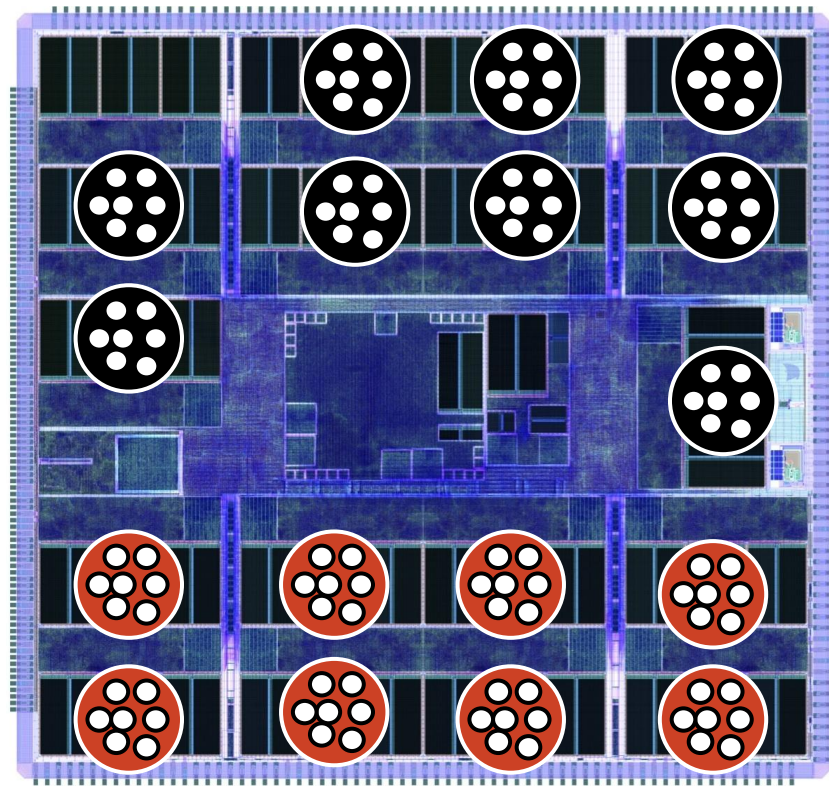
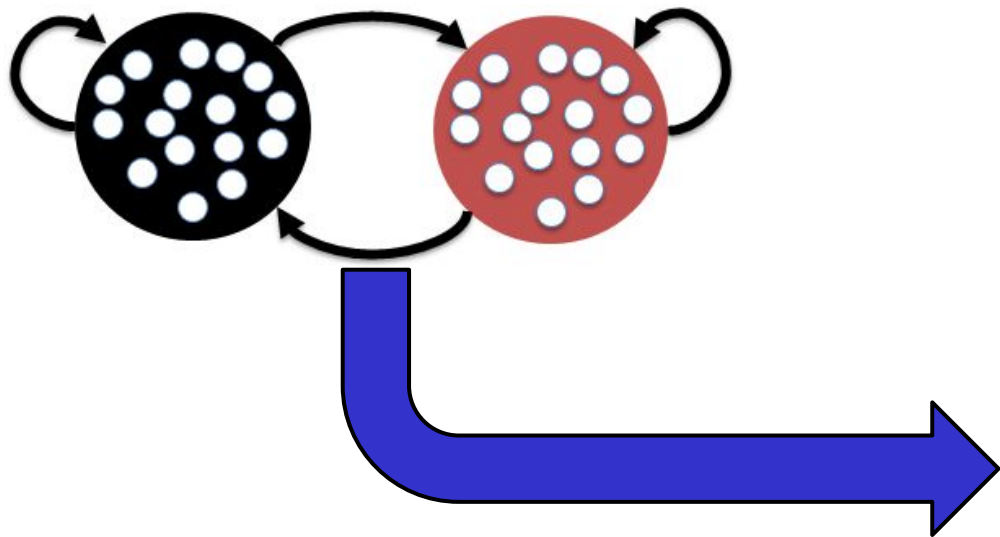
$\Delta w = \Delta a (w - w_{\min})$ if $\Delta a < 0$ (Depression)

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

Running on SpiNNaker

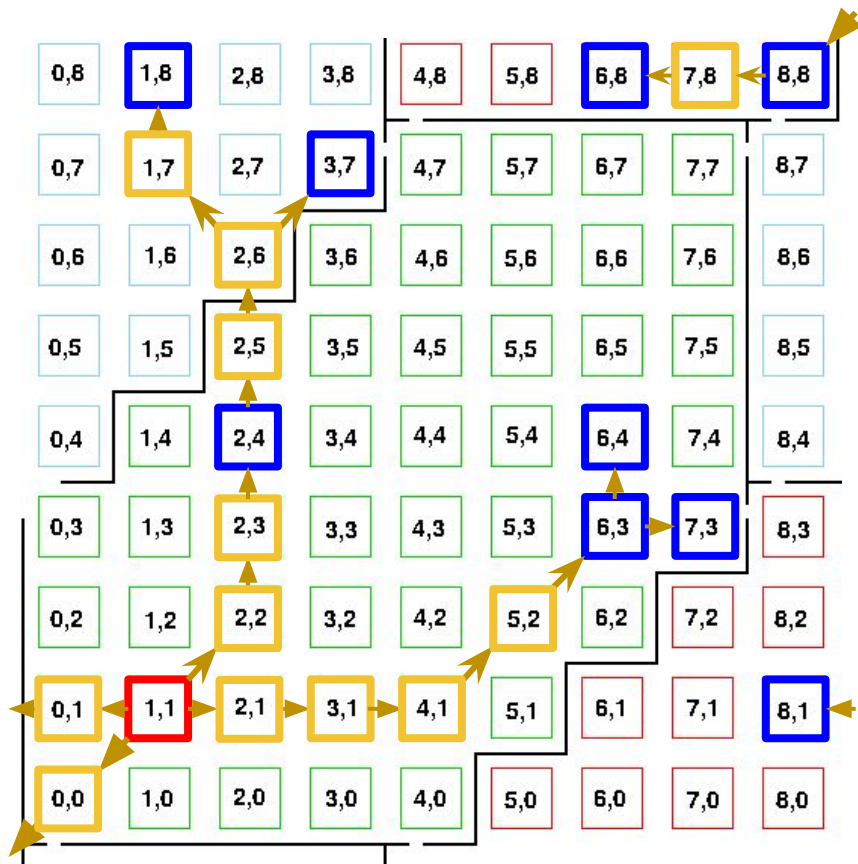
PyNN - Run

`p.run(100)`



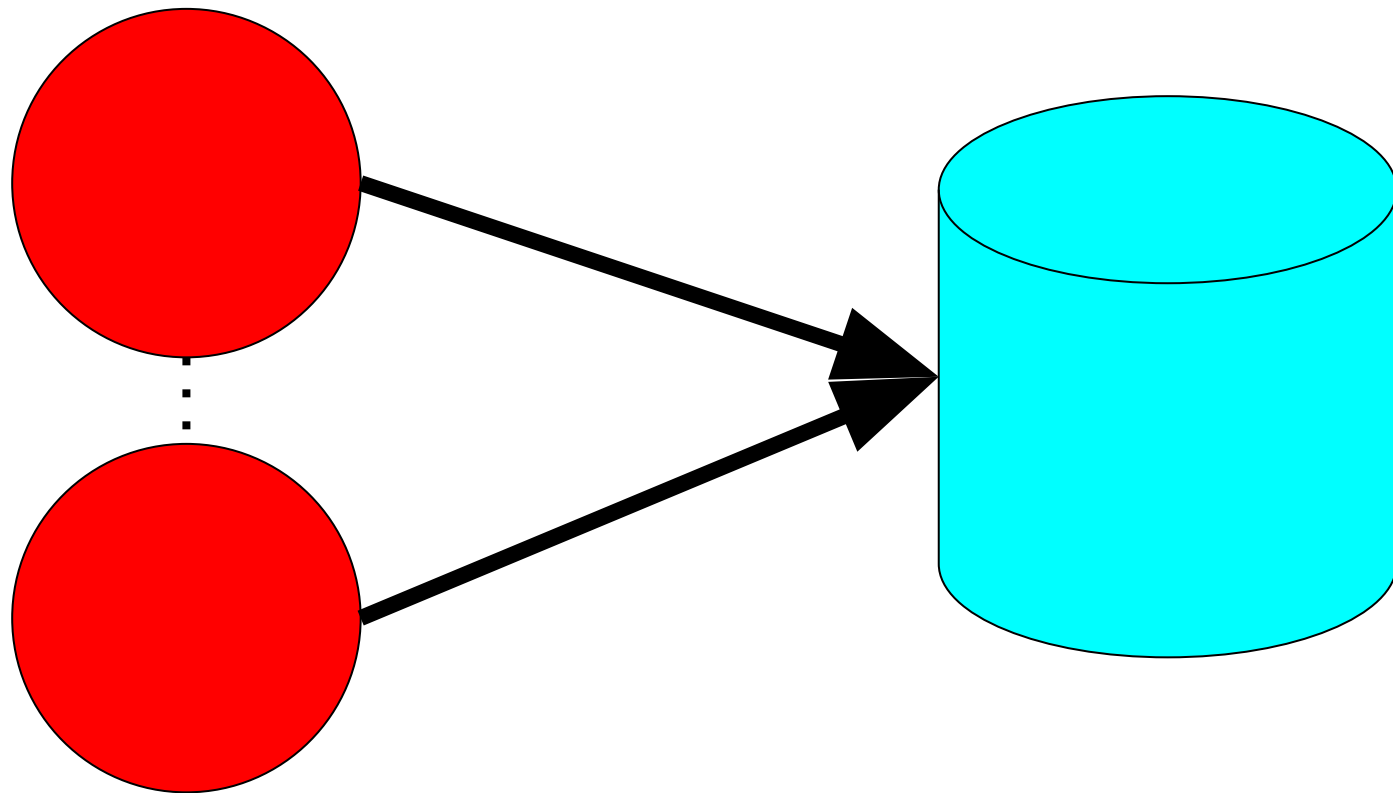
PyNN - Run

p.run(100)



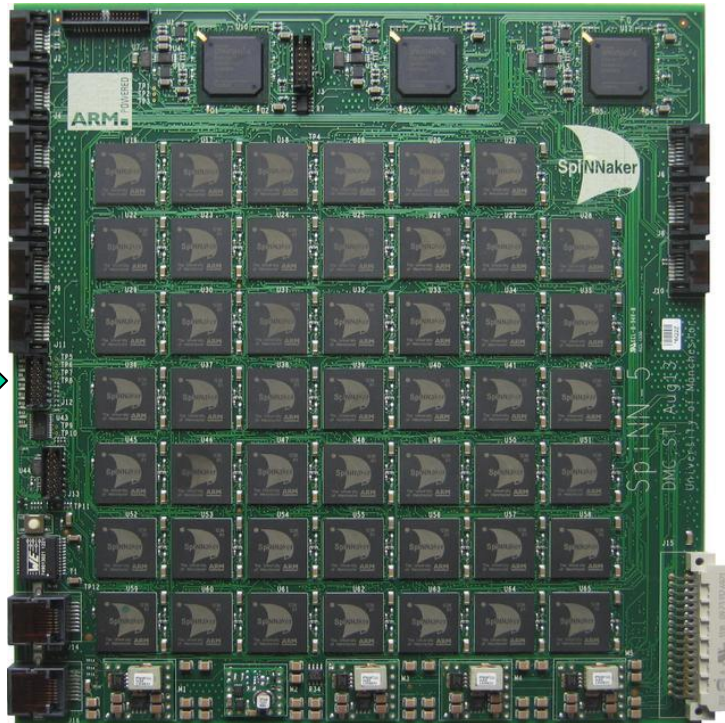
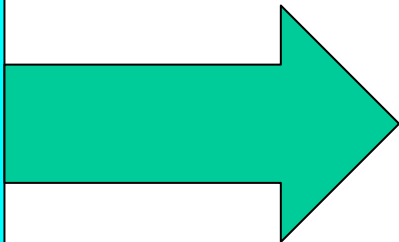
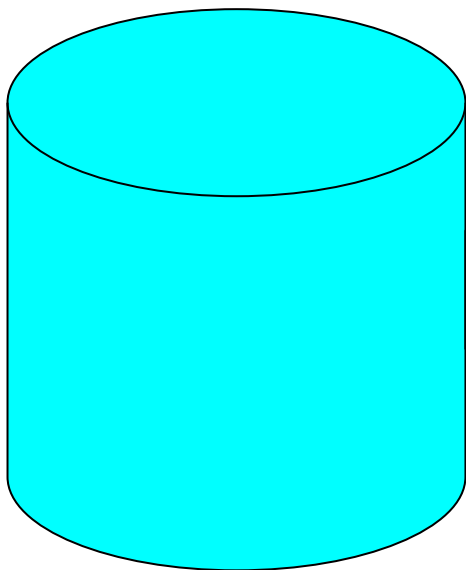
PyNN - Run

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



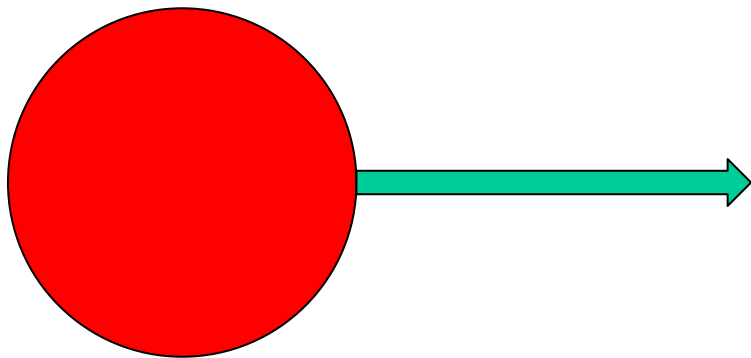
PyNN - Run

`p.run(100)`



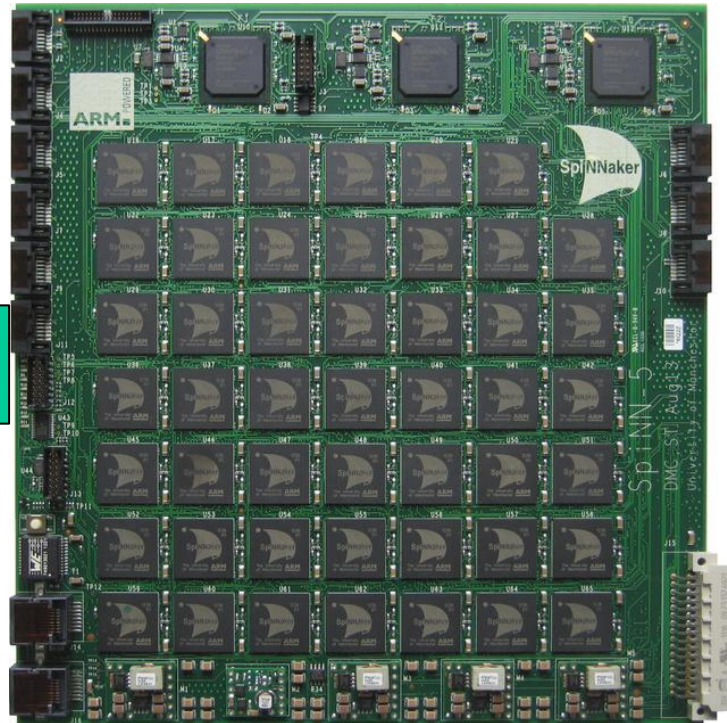
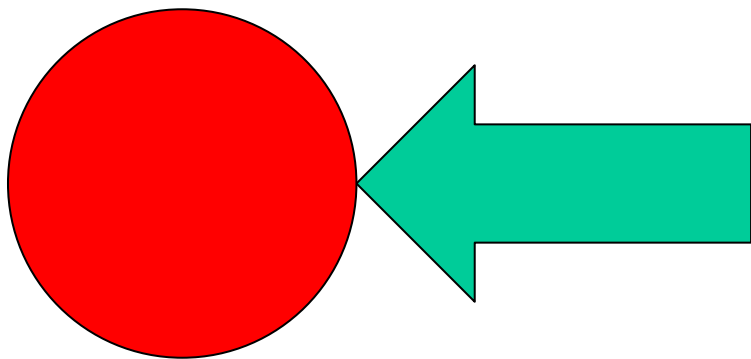
PyNN - Change and Run Again

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



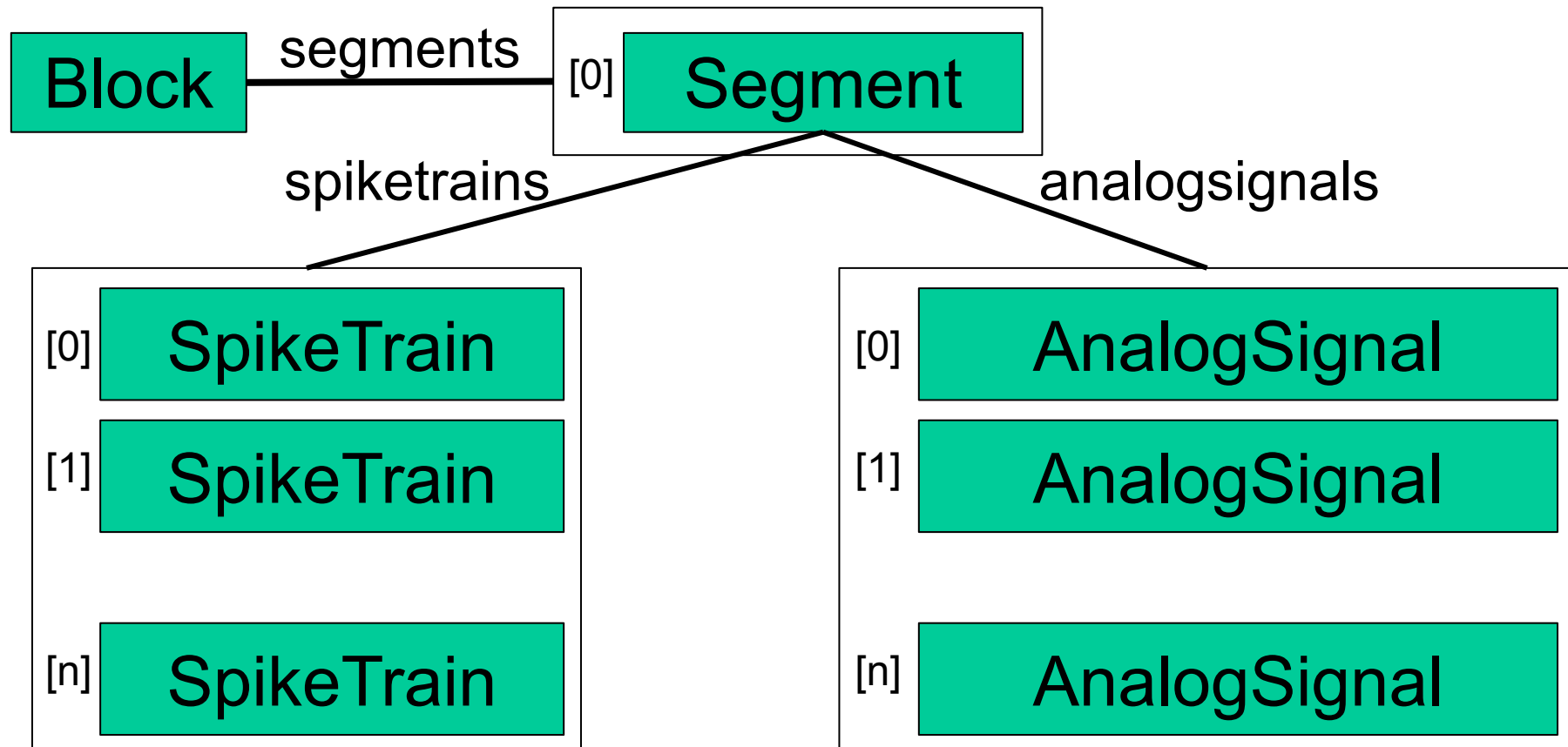
PyNN - Get Data

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

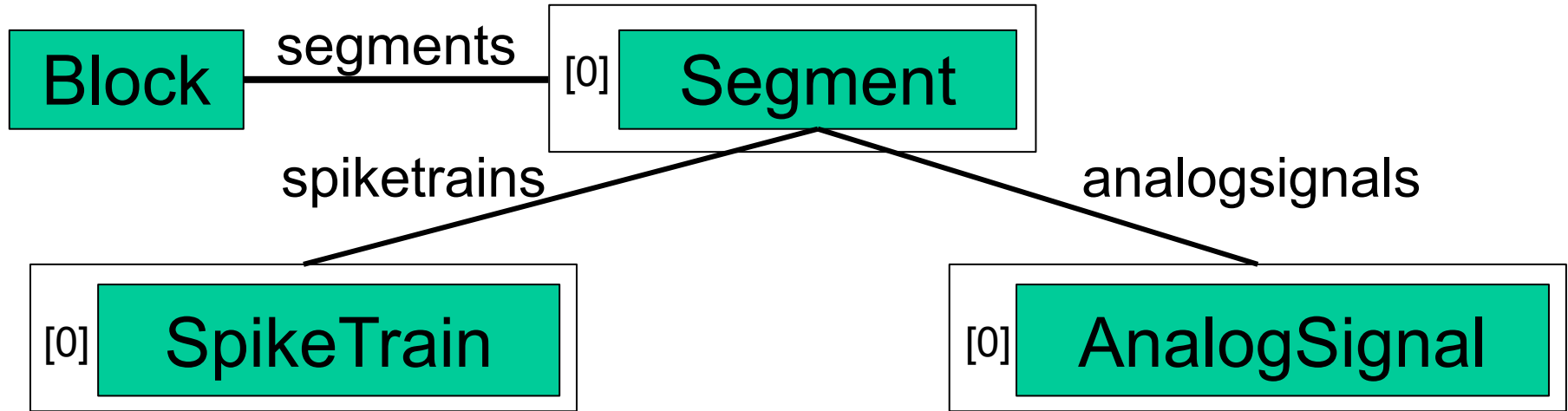


Reading Results

PyNN - Neo Data



PyNN - Neo Data



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

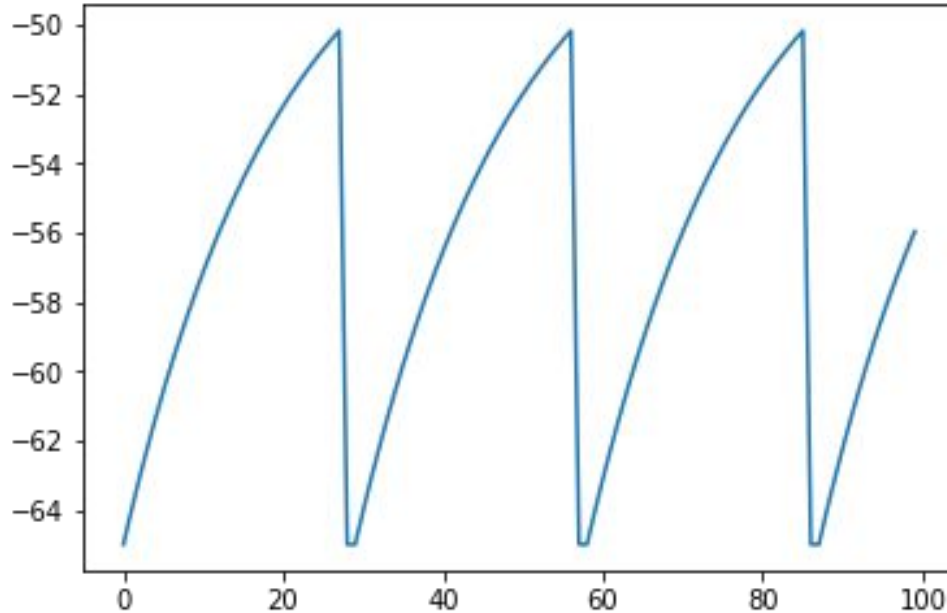
PyNN - Plotting

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

```
plt.figure()
```

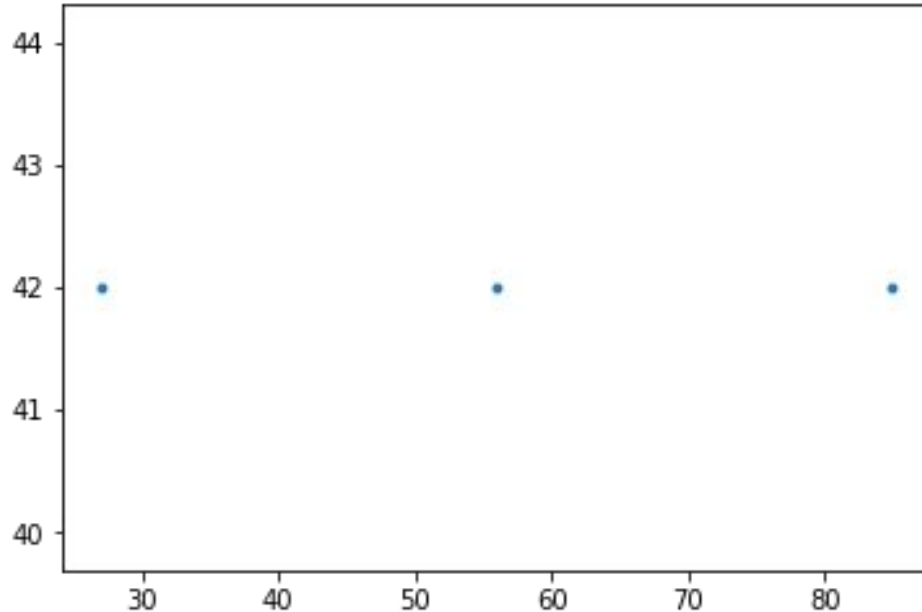
```
plt.plot(v[0].times, v[0])
```

```
plt.show()
```



PyNN - Plotting

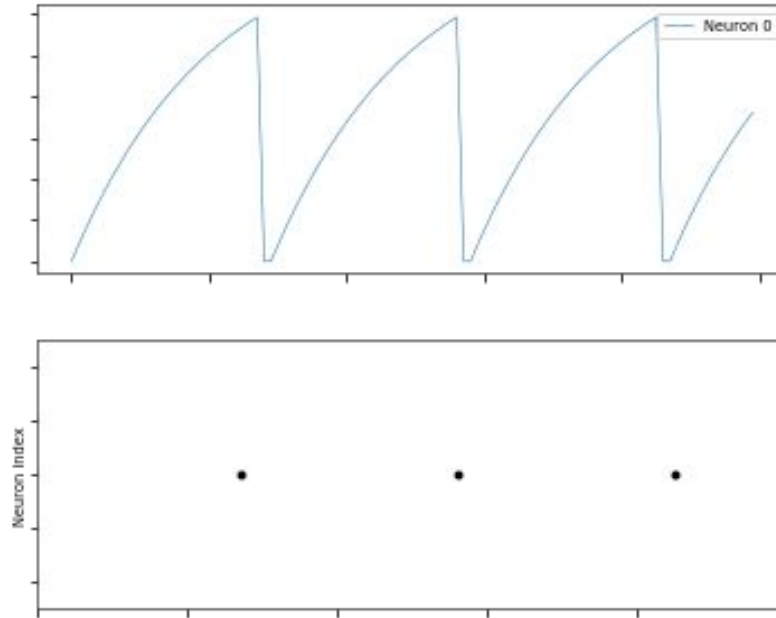
```
plt.figure()  
y = [1 for i in range(len(spikes[0]))]  
plt.plot(spikes[0], y, ".")  
plt.show()
```



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

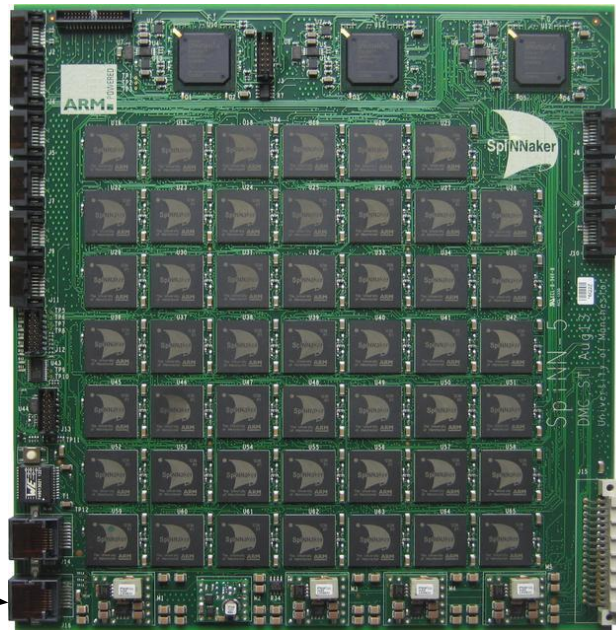
send_spikes

SpikeInjector
label="injector"

send_spikes("injector", [0])

SpynnakerLiveSpikesConnection
send_labels=["injector"]

Multicast Key(s)
(Spike)

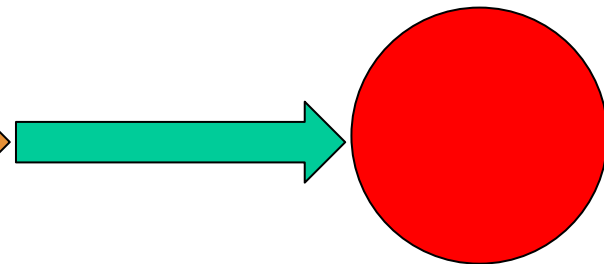


Input from Environment: Rates

add_poisson_live_rate_control

set_rates

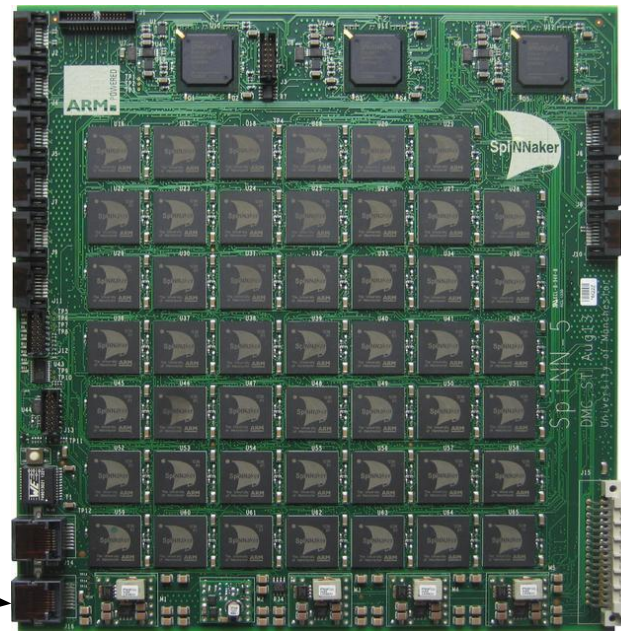
SpikeSourcePoisson
label="input"



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

SpynakerPoissonControlConnection(
poisson_labels=["input"])

Multicast Key(s)
and Payload(s)



Output to Environment: Spikes

activate_live_output_for

recv

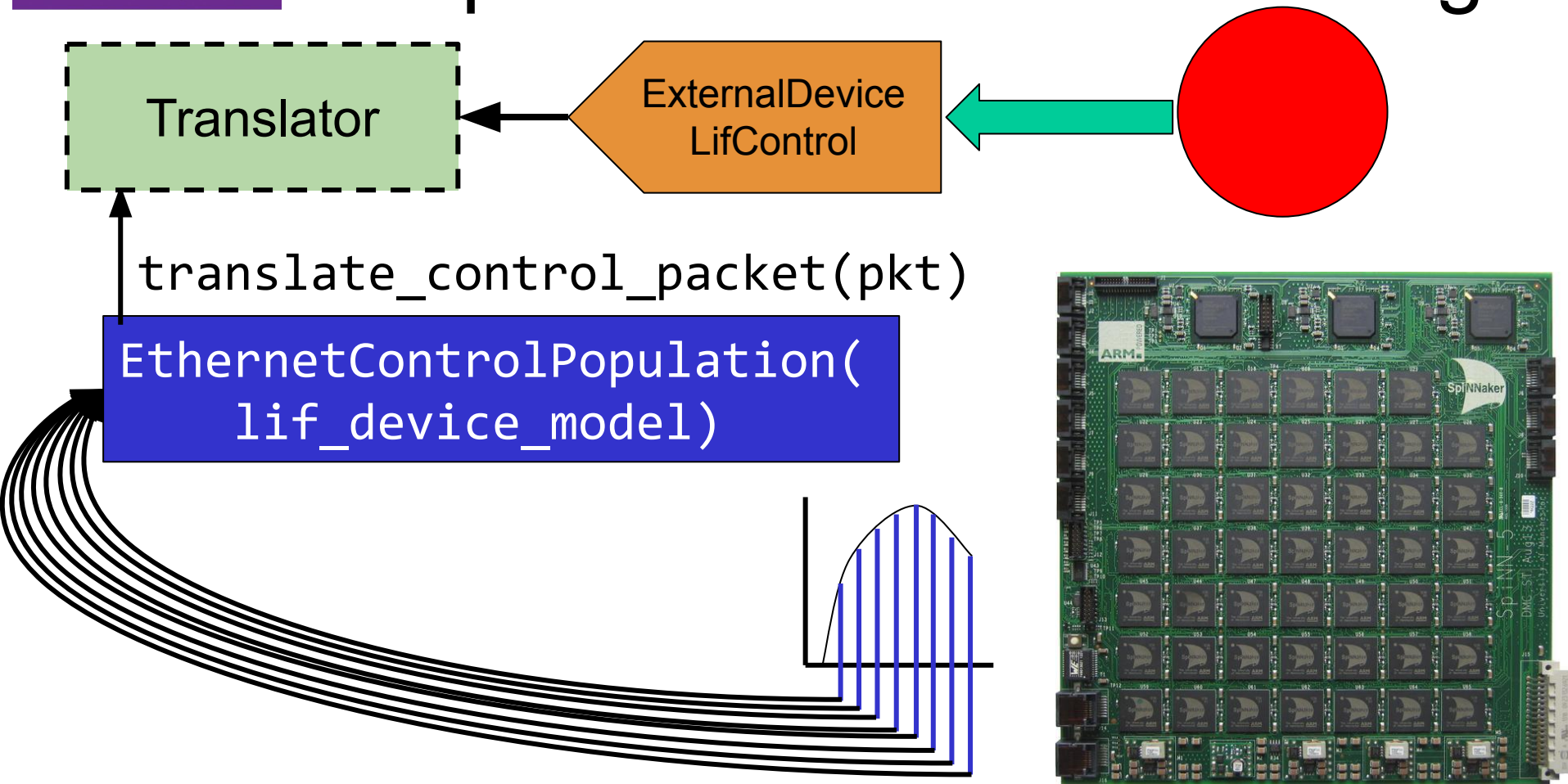
recv("pop", time, neuron_ids)

SpynnakerLiveSpikesConnection(
receive_labels=["pop"])

Multicast Key(s)
(Spike)



Output to Environment: Voltage



SpiNNaker Tutorials

The screenshot displays the JupyterLab web interface. The browser address bar shows the URL: `lab.jsc.ebrains.eu/user/rowley/lab/tree/shared/Andrew%20Test%20Collab/SpiNNakerJupyterE...`. The interface is divided into two main panels.

Left Panel (File Browser): Shows a directory tree for `Andrew Test Collab / SpiNNakerJupyterExamples /`. The files and folders listed are:

Name	Last Modified
00.Setup	seconds ago
01.RunningPyNNSimulations	seconds ago
02.LiveInputAndOutput	seconds ago
Integration tests	seconds ago
spinnaker_jupyter_examples	seconds ago
LICENSE	seconds ago
pyproject.toml	seconds ago
README.md	seconds ago
setup.cfg	seconds ago
setup.py	seconds ago

The file `01.RunningPyNNSimulations` is circled in red.

Right Panel (Launcher): Shows the path `shared/Andrew Test Collab/SpiNNakerJupyterExamples` and a section titled "Notebook". It displays a grid of available kernels:

- Python 3 (ipykernel)
- EBRAINS_release_v0.1_202109
- EBRAINS-22.07
- EBRAINS-22.10
- EBRAINS-23.02
- EBRAINS-23.06
- EBRAINS-23.09
- EBRAINS-24.04
- EBRAINS-experimental

The bottom status bar shows "Simple" mode, a memory usage of 256.77 / 2048.00 MB, and a "Launcher" button.

SpiNNaker Tutorials on Jupyter

<https://lab.ebrains.eu/>

[https://github.com/SpiNNakerManchester/
SpiNNakerJupyterExamples](https://github.com/SpiNNakerManchester/SpiNNakerJupyterExamples)

01. Running PyNN Simulations

02. Live Input And Output

PyNN Documentation:

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