

# Providing community access to large-scale neuromorphic computing systems

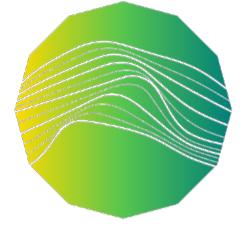
M. Sénoville<sup>1</sup>, H. Agigli<sup>1</sup>, O. Ates<sup>1</sup>, J. Duperrier<sup>1</sup>, D. Guarino<sup>1</sup>, B. Lungsi Sharma<sup>1</sup>, E. Müller<sup>2</sup>, A.G.D. Rowley<sup>3</sup>, A.P. Davison<sup>1</sup>  
and the EBRAINS Neuromorphic Computing Collaboration

<sup>1</sup>Paris-Saclay Institute of Neuroscience, UMR 9197, Centre National de la Recherche Scientifique/Université Paris-Saclay, France

<sup>2</sup>Kirchhoff Institute for Physics, Heidelberg University, Heidelberg, Germany

<sup>3</sup>Advanced Processor Technologies Group, Department of Computer Science, University of Manchester, Manchester, United Kingdom

## The EBRAINS Neuromorphic Computing Platform



EBRAINS Research Infrastructure offers access to two unique large-scale neuromorphic computing systems :

### BrainScaleS : the physical model machine

Location : Heidelberg (Germany)



20 wafer modules  
Local analogue computing  
4 million neurons  
1 billion plastic synapses  
Binary, asynchronous communication  
running at x1000 real-time



### SpiNNaker : the many-core machine

Location : Manchester (UK)

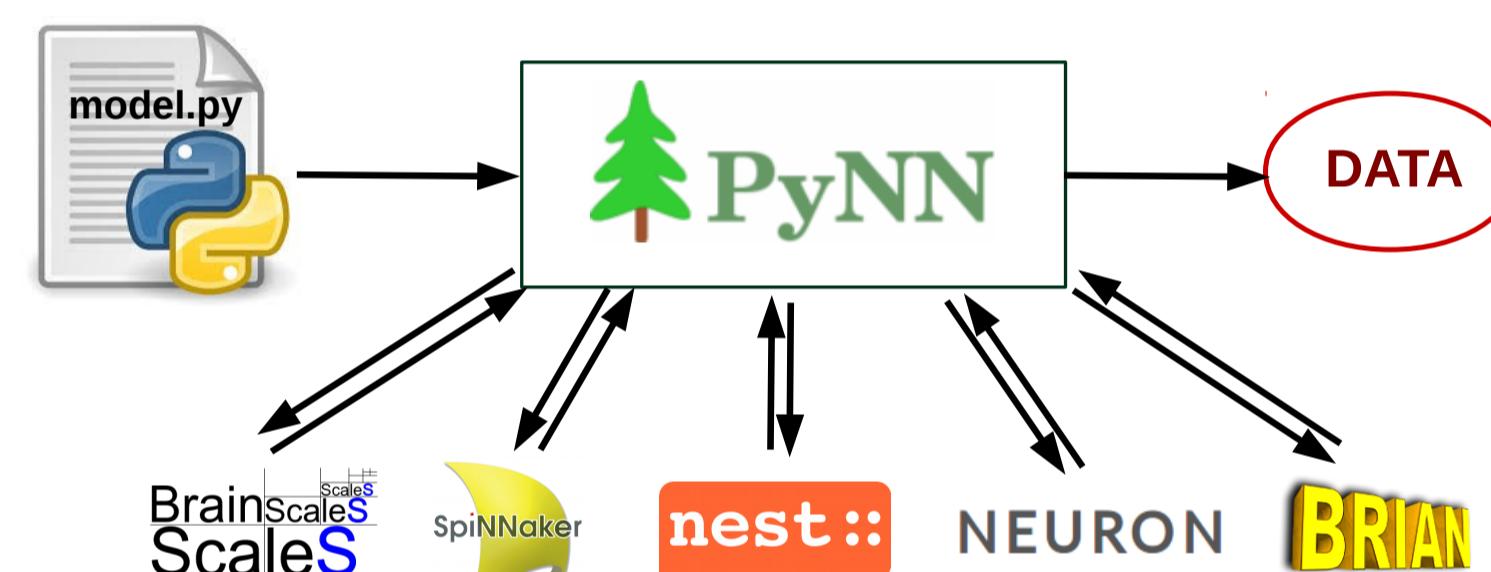


over 1 million cores  
up to 1 billion neurons  
up to 1 trillion synapses  
address-based, small packet,  
asynchronous communication  
real-time simulation



## PyNN : a common API for neuronal network modelling

- ▷ facilitate model sharing and reuse
- ▷ simplify validation of simulation results
- ▷ provide a common platform on which to build other tools
- ▷ provide a more powerful API for neuronal network modelling
- ▷ hide complexity of parallelization from user



```
import pyNN.nest as simulator
import pyNN.neuron as simulator
import pyNN.brian as simulator
import pyNN.brainscales as simulator
import pyNN.spiNNaker as simulator

cell_type = ...

p1 = simulator.Population(size1, cell_type, structure)
p2 = simulator.Population(size2, another_cell_type, structure)

all = p1 + p2
all.record(["spikes", "v"])

connections = simulator.Projection(p1, p2, connection_rule, synapse_type)

simulator.run(1000.)
```

PyNN is a **simulator-independent** language for building spiking neuronal network models and provides a library of :

- ▷ standard neuron, synapse and synaptic plasticity models
- ▷ commonly-used connectivity algorithms

The user is not restricted to the standard models and can also use any neuron or synapse model supported by your simulator.

For downloads and full documentation see :

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



## Request free test access

- ▷ To receive support and advice, visit  
<https://ebrains.eu/support/>

<https://ebrains.eu/service/neuromorphic-computing/>

## Access through the Collaboratory

The screenshot shows the EBRAINS Collaboratory Job Manager interface. It displays a list of submitted jobs for the BrainScaleS and SpiNNaker systems. Each job entry includes the ID, status (e.g., submitted, error), system, code URL, submission date, and submitted by user. Below the main list, there is a detailed view for job 153802, showing its status as finished, command, hardware config, provenance, and log files.

The Neuromorphic Job Manager is a web based app providing a graphical interface for submitting jobs to the BrainScaleS and SpiNNaker systems and for retrieving job results, with detailed provenance metadata including full log files.

As a Community app the Job Manager is integrated into the EBRAINS Collaboratory, which aims to be a workspace for scientific collaboration, sharing and collaboration around data, software and services.

The screenshot shows the EBRAINS Neuromorphic Computing Service: Job Manager interface. It includes sections for selecting a hardware platform (SpiNNaker), choosing a simulation platform, and entering code via an editor or file upload. A command line input field is also present. The right side of the interface shows optional configuration options and a summary of the job setup.

## Access through Jupyter Notebooks

The screenshot shows a Jupyter Notebook interface running on the BrainScaleS system. The code cell contains Python code for a chain test, including imports, configuration parameters, and execution commands. The output cell shows the results of the run, including logs and the creation of a 'chain\_data.pkl' file. The notebook interface includes tabs for different cells and a toolbar for file operations.

Batch mode via a Jupyter Notebook

- ▷ in the EBRAINS Collab ("Lab")
- ▷ on local computer

Requires the use of a **Python Client** : the model is sent as file to the neuromorphic systems, then executed and the results are sent back as files.

Installing the Python client

```
$ pip install hbp_neuromorphic_platform
```

Using the Python client

```
In [1]: import nmpi
In [2]: c = nmpi.Client("myusername")
In [3]: job_id = c.submit_job(source="/path/to/my/code/PyNN_script.py",
                           platform=nmpi.BRAINTSCALES,
                           collab_id="collab-name")
Out[3]: Job submitted

In [4]: c.job_status(job_id)
Out[4]: 'queued'

In [5]: job = c.get_job(job_id, with_log=True)

In [6]: c.download_data(job, local_dir="/path/for/download")
Out[6]: ['/path/for/download/job_123/sim_results.png', '/path/for/download/job_123/reports.zip']
```

An interactive mode is also available on :

- ▷ BrainScaleS : <https://juphub.bioai.eu/>
- ▷ SpiNNaker : <https://spinn-20.cs.man.ac.uk/>



Human Brain Project

Fraunhofer IZM

CITEC

TECHNISCHE  
DRESDEN

CNRS

TU  
Graz

University of  
Hertfordshire

UH

Sabancı  
Universitesi

EPFL

Middlesex  
University

MANCHESTER

KTH

US  
UNIVERSITY OF SUSSEX

DEUTSCHER  
VETERTIN

COLLEGE

Co-funded by  
the European Union



This research has received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 945539 (HBP SGA3), 785907 (HBP SGA2), 720270 (HBP SGA1) and the FP7 grant number 604102 (HBP RUP).