EBRAINS modelling workflows implementation status at M9 (D5.2 – SGA3)

M9 status on implementation procedures of interconnecting tools according to EBRAINS architecture specification, workflows and simulation engines, and mechanisms to port established SGA2 use cases into EBRAINS.

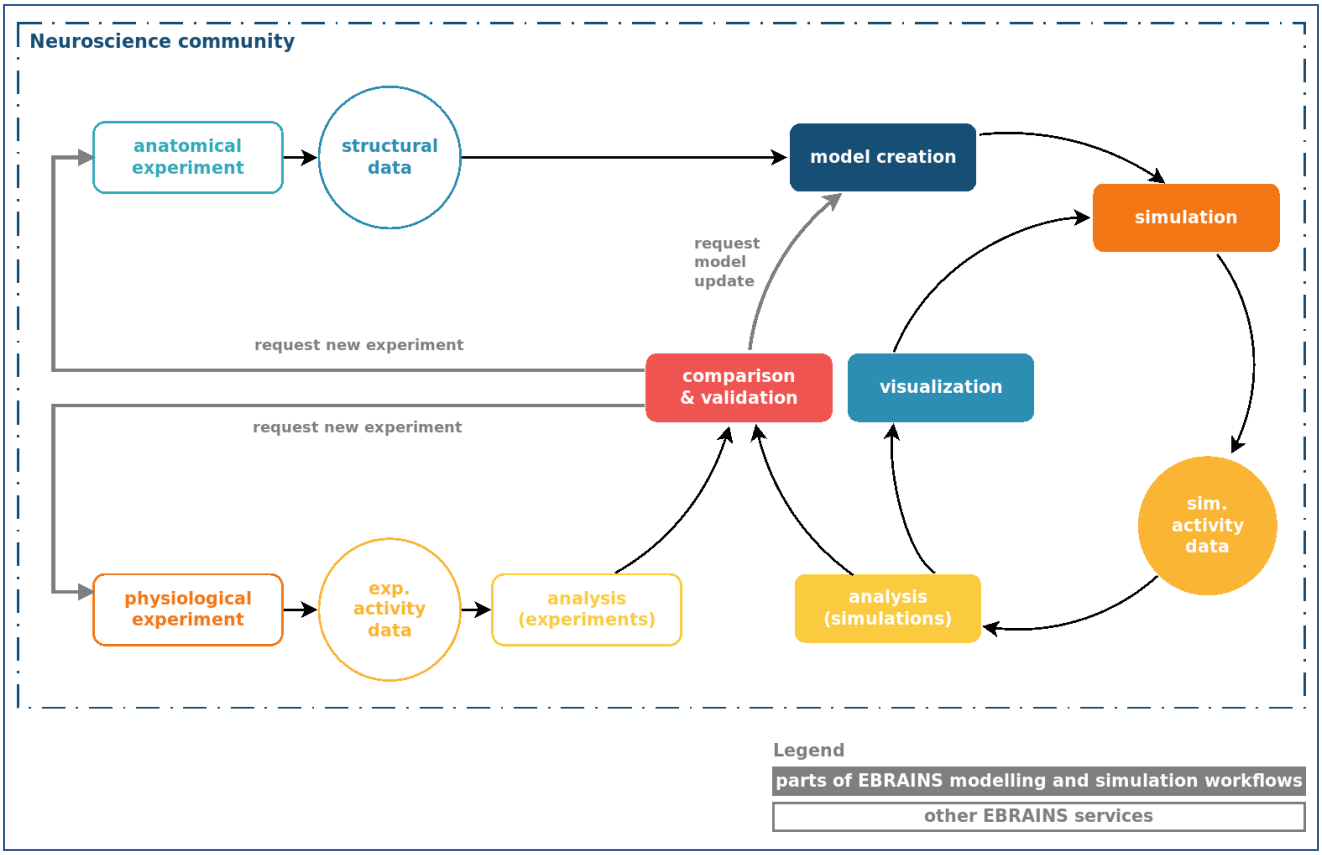


Figure 1: Modelling and simulation life cycle

EBRAINS modelling and simulation services provide the tools to create and investigate computational models of the brain. These tools and services include web front ends, software libraries and applications for model building, parameter estimation, simulation, data analysis and visualisation (filled boxes), which can be combined to create workflows for the interpretation of experimental data on brain activity and structure (outlined boxes).

Project Number:	945539	Project Title:	HBP SGA3

Document Title:	EBRAINS modelling workflows implementation status at M9
Document Filename:	D5.2 (D49) SGA3 M9 ACCEPTED 210303.docx
Deliverable Number:	SGA3 D5.2
Deliverable Type:	Other
Dissemination Level:	PU = Public
Planned Delivery Date:	SGA3 M9 / 31 Dec 2020
Actual Delivery Date:	SGA3 M10 / 18 Jan 2021; Accepted 3 Mar 2021
Author(s):	Andrew DAVISON, CNRS (P10)
Compiled by:	Andrew DAVISON, CNRS (P10), Anne ELFGEN, JUELICH (P20)
Contributor(s):	Andrew DAVISON, CNRS (P10), contributed to all sections Susanne KUNKEL, NMBU (P44), contributed to sections 1, 2, and 5; overall review Anne ELFGEN, JUELICH (P20), overall review Jeanette HELLGREN KOTALESKI, KTH (P39) Rebecca WADE, HITS (P26) Michele MIGLIORE, CNR (P12) Felix SCHÜRMANN, EPFL (P1) Abigail MORRISON, JUELICH (P20) Markus DIESMANN, JUELICH (P20) Sandra DIAZ, JUELICH (P20) Huifang WANG, AMU (P78) Wouter KLIJN, JUELICH (P20) Lena ODEN, JUELICH (P20) Sonja GRÜN, JUELICH (P20) Michael DENKER, JUELICH (P20) Benjamin WEYERS, UT (P130) Óscar David ROBLES SÁNCHEZ, UPM (P68) Giulia ROSSETTI, JUELICH (P20) Dennis TERHORST, JUELICH (P20) Jean-Denis COURCOL, EPFL (P1) James KING, EPFL (P1) Werner VAN GEIT, EPFL (P1) Johannes HJORTH, CNRS (P10) Robin Gilbert DE SCHEPPER, UNIPV (P70) Adam HOSPITAL, IRB Barcelona Gaute Einevoll, NMBU (P44) Espen HAGEN, NMBU (P44) Torbjørn Vefferstat NESS, NMBU (P44) Stefano ANTONEL, EPFL (P1) Luca Leonardo BOLOGNA, CNR (P12) Brent HUISMAN, JUELICH (P20) Katrien VAN LOOK, EPFL (P1)
WP QC Review:	Anne ELFGEN, JUELICH (P20)
WP Leader / Deputy Leader Sign Off:	Susanne KUNKEL, NMBU (P44)
PCO QC Review:	Guy WILLIS, EPFL (P1)
Description in GA:	M9 status on implementation procedures of interconnecting tools according to EBRAINS architecture specification, workflows and simulation engines, and mechanisms to port established SGA2 Use Cases into EBRAINS.
Abstract:	This report presents the status of tools and workflows for modelling and simulation in EBRAINS, as of late 2020. It contains a brief guide to each of the available and planned tools/workflows with links to further information, a list of recent advances, and a summary of future plans.
Keywords:	Modelling, simulation, data analysis, model validation, visualisation, molecular, subcellular, single neuron, network, whole brain, software libraries, web apps, workflows
Target Users/Readers:	Computational neuroscience community, computer scientists, consortium members, funders, general public, HPC community, neuroinformatics specialists, neuroscience community, neuroscientists, platform users, researchers, students.

Table of Contents

Table of Tables

Table of Figures

1. Overview

Modelling and simulation are essential instruments in our endeavour to understand the human brain. Computer simulations allow us to develop and investigate different models of neuronal systems, to integrate and combine various experimental findings in the models, and to test and challenge theories about brain function and dysfunction in view of experimental findings. In this way, simulation acts as the main conduit for interchange between experiment and theory.

Neuroscientists have different views on how to best tackle the intricacy of complex dynamic systems such as the brain and therefore make use of a broad range of models, which differ in size, complexity and level of detail. EBRAINS modelling and simulation services comprise simulation engines operating at different abstraction levels to cover the entire modelling scope, ranging from molecular/subcellular to single neuron to network to whole brain. To account for the multi-scale architecture of the brain, co-simulation of models of adjoining scopes by concerted use of multiple simulators is an integral part of the offered simulation services.

The complexity and versatility of the brain, and the variations from one brain to another, are major scientific challenges, driving not only the development of simulation technology but also the development of tools for model creation, model validation, data analysis, and visualisation. As part of the EBRAINS modelling and simulation services, such tools embed the simulation engines to form integrated modelling and simulation workflows (Figure 1), thereby fostering reproducible research and helping create a quality user experience.

Here, we present recent progress regarding the integration of the tools with the EBRAINS research infrastructure, which is also reflected in the presentation of the tools on http://ebrains.eu/services/simulation evolving towards a coherent ensemble. This includes, for example, the migration of the tools from the original Human Brain Project (HBP) authentication/authorization system to the EBRAINS system and the hosting of the services on the EBRAINS virtual machine service.

EBRAINS modelling and simulation services will usher in a new era of computational neuroscience research by offering the technical solutions for brain researchers to conduct sustainable simulation studies, build upon prior work, and the means to share their results.

2. Introduction

2.1 The roles of modelling and simulation in neuroscience and in AI

Modelling and simulation are powerful tools for understanding neural systems such as the brain, for integrating and interpreting data from different sources, and for developing novel, bio-inspired approaches to computation and artificial intelligence.

2.2 Challenges

The rate and resolution at which new data about the brain can be acquired, the precision of experimental manipulations, and the complexity of experimental protocols have all exploded in recent years, as technologies such as Neuropixels probes, 3D-polarized light imaging, ultra-high field fMRI, optogenetics, and machine-learning-supported image and data analysis have become available.

This has brought new challenges to brain modelling and simulation: a need for increased collaboration and, with this, for clear and efficient communication and sharing of models, theories, and results; a need to make efficient use of massively parallel, high performance computing (including neuromorphic computing systems); a need to simulate neural systems at multiple scales simultaneously; a need to systematically compare simulation results to experimental data; and a

need to educate and train students, scientists and engineers to understand and use new, wide-ranging and multidisciplinary skills and concepts.

2.3 The EBRAINS response: tools, workflows and services

In response to these challenges, the HBP has conceived an integrated set of sophisticated digital tools and services to create and investigate mathematical models of the brain which are currently being implemented on the EBRAINS platform[1]. The tools and services available on EBRAINS to date are accessible via https://ebrains.eu/services/simulation.

These tools and services, built upon the EBRAINS hardware infrastructure and middleware, include web front ends, software libraries and applications for model building, parameter estimation, simulation, and data analysis. They can be combined to create workflows, and EBRAINS provides both pre-configured workflows for common use cases and the ability to flexibly construct new workflows (Figure 1).

Some of the tools and services are mature, others are available as prototypes, others are not yet available, but are planned to become available within the next two years.

Tools and services can be categorised by type (web app, software library, command-line application, web service), model scope (molecular and subcellular mechanisms, single neurons, networks, whole brain), by abstraction level (statistical modelling, systems biology, point neuron models, biophysical neuron models, population modelling, etc.), or by their role in the modelling and simulation life cycle (Figure 1). Below we give a brief introduction to the available and planned tools and services, grouping them by life cycle role, and using the icons shown in Table 1 to indicate type, model scope, abstraction level, and the availability/maturity level. Where tools and services have multiple features with different levels of maturity, the icon indicates the most mature features.

Table 1: Legend for icons

Type		Availability/maturity		Scope		Modelling abstraction level	
web app		mature, available now (TRL 7-9)		molecular		nanoscale	**NS**
command-line tool or library		prototype available (TRL 5-6)		subcellular		systems biology	**SB**
		in progress / future work		single neurons		point neuron models	**PT**
				networks		biophysical models	**BI**
				whole brain		population models	**PP**
						statistical models	**ST**

2.4 Building models

Tool or service	Characteristics	Modelling abstraction level	Purpose
ArDOCK		**NS**	Detect potential interaction sites on the surface of a protein.

[1] https://ebrains.eu

Tool or service	Characteristics	Modelling abstraction level	Purpose
Molecular level toolset		NS	Set up, run and analyse molecular level calculations, e.g. tauRAMD calculations to compute dissociation constants.
Subcellular webapp		SB	Create and simulate brain molecular networks.
Subcellular model building toolset		SB	Construct, simulate and validate subcellular models.
Feature Extraction Tool		BI	Extract a wide variety of electrophysiological features from experimental recordings.
Hodgkin-Huxley Neuron Builder		BI	Optimise a single cell model against experimental data on HPC resources.
Electrical model building toolset		BI	Build and validate biophysically detailed electrical neuron models.
PyNN		PT BI	Build simulator-independent spiking neuronal networks and simulation experiments.
NESTML		PT	Build simulator-independent point neuron and synaptic plasticity models.
Snudda		BI	Build networks of biophysically-detailed neurons where the connectivity is based on the neuron morphologies.
Brain Scaffold Builder		BI	Rapidly construct biologically realistic network models.
TVB		PP	Construct and simulate personalised brain network models.

2.5 Running simulations

Tool or service	Characteristics	Modelling abstraction level	Purpose
CGMD Platform		NS	Prepare, run and analyse coarse-grained molecular dynamics simulations.
SSB Platform / METNET		NS SB	Predict the downstream effects upon ligand binding/disease associated variants such as protein/metabolite concentrations/dose response curves, and explore metabolic networks using automated bio-mathematical models.
MoDEL-CNS		NS	Access atomistic-MD trajectories for relevant signal transduction proteins in a web browser.
SDA		NS	Simulate diffusional association of proteins. Compute bimolecular association rate constants.
Single Cell *In Silico* Experiment Tool (BlueNaaS)		BI	Neuron as a Service: Simulate and visualize detailed single neuron models in the web browser.
Small Circuit *In Silico* Experiments Tool		BI	Run simulation experiments with a small hippocampal neural circuit.

Tool or service	Characteristics	Modelling abstraction level	Purpose
Brain Areas Circuit *In Silico* Experiments Tool		BI	Run simulation experiments with a full-scale model of the hippocampus CA1 region.
NEST		PT	Run highly-scalable simulations of point neuron networks.
Arbor		BI	Run scalable, high-performance simulations of multi-compartment neurons in large networks.
NEURON/CoreNEURON		BI	Run scalable, high-performance simulations of multi-compartment neurons in large networks in the well-established NEURON simulation environment.
Neuromorphic Computing Job Manager app		PT	Run simulation experiments with point neuron networks on the BrainScaleS and SpiNNaker neuromorphic computing systems.
Multiscale co-simulation framework		PT BI PP	Connect simulation tools at runtime to enable multiscale co-simulations.
TVB		PP	Simulate personalised brain network models locally, on the Collaboratory, or directly on HPC for large parameter explorations.

2.6 Data analysis

Tool or service	Characteristics	Modelling abstraction level	Purpose
Neo		SB PT BI PP ST	Load, save, annotate and manipulate electrophysiology and imaging data from experiment and simulation in a standardised framework.
Elephant		SB PT BI PP ST	Analyse neuronal activity data from experiment and simulation using high-performance, well-tested methods.
LFPy		BI PP	Calculate electric and magnetic brain signals from multi-compartment neuron models and networks.
TVB		PP	Load, analyse, visualise and explore experimental and simulated brain activity time series.

2.7 Visualisation

Tool or service	Characteristics	Modelling abstraction level	Purpose
SimVisSuite		PT BI PP	Interactively visualise and analyse network and neuron-level simulations.
NEST Desktop		PT	Build, run, visualise and analyse simulations with the NEST simulator in a web-based GUI.

Tool or service	Characteristics	Modelling abstraction level	Purpose
SimDaVis/Insite		SB PT BI PP ST	Continuously access simulation data at runtime.
Elephant visualisation		SB PT BI PP ST	Visualise analysis results in Jupyter notebooks.

2.8 Validating models against experimental data

Tool or service	Characteristics	Modelling abstraction level	Purpose
Validation test libraries		SB PT BI PP ST	Develop and run structured, quantitative model validation tests.
Model Catalog		SB PT BI PP ST	Explore and visualise computational models, validation tests and validation results.

3. Current status, recent changes, future plans

3.1 Building models

3.1.1 ArDOCK

Type: web app
Scope: molecular
Abstraction level: nanoscale
Status: Available for use
Homepage & more information: https://ardock.ibcp.fr/

ArDOCK is a web tool for "arbitrary docking" to detect interaction sites on proteins (Reille *et al.* 2018[2]). Arbitrary docking can reveal potential interaction sites on the surface of a protein using docking with a set of 25 random protein "probes". It has been shown that the random probes interact in a non-random manner on protein surfaces, and that targeted regions are enriched in biological interfaces. Docking is performed by the Hex software, using spherical polar Fourier Correlations. Figure 2 shows a screenshot of the ArDOCK web tool.

2 Reille S, Garnier M, Robert X, Gouet P, Martin J, Launay G (2018) Identification and visualization of protein binding regions with the ArDock server. Nucleic Acids Res. 46(W1):W417-W422. (P1372)

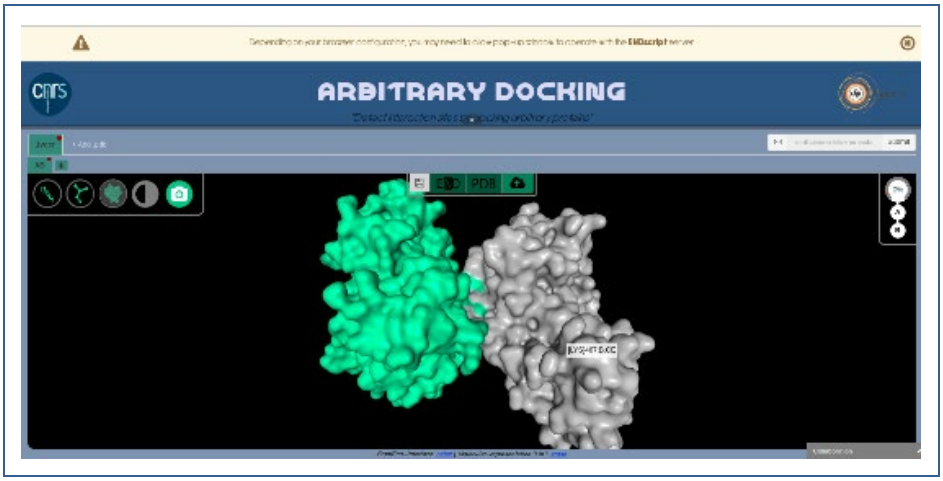


Figure 2: Screenshot of the ArDOCK webserver for the prediction of potential interaction sites on proteins

3.1.1.1 Outlook

This component of the tool set developed in the HBP Collaboratory v1 will be made available to the new EBRAINS platform.

3.1.2 Molecular level tool-set

Type: workflows/scripts/tools/apps
Scope: molecular
Abstraction level: nanoscale
Status: Partially available on HBP Collaboratory v1
Homepage & more information: https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/molecular_level/molecular_level.html

This is a set of workflows (Jupyter Notebooks and scripts) and other tools that can be used to generate parameters and constraints for subcellular modelling from using information on the three-dimensional structures of biomolecules. A range of different types of calculations can be done. Figure 3 shows an illustration of a tauRAMD (Kokh *et al*. 2018[3]; Kokh *et al*. 2019[4]) workflow which is used to compute relative dissociation rate constants for a set of compounds from a protein. The workflow is described on kbbox[5] (Bruce *et al*. 2019[6]) and a Jupyter Notebook is being developed for the EBRAINS platform. The workflows in the toolset will also make use of MD-IFP[7] (Kokh *et al*. 2020[8]), a Python workflow for the generation and analysis of Protein-ligand Interaction Fingerprints from Molecular Dynamics trajectories. The workflows will also use data in molecular level resources like

[3] Kokh DB, Amaral M, Bomke J, Grädler U, Musil D, Buchstaller HP, Dreyer MK, Frech M, Lowinski M, Vallée F, Bianciotto M, Rak A, Wade RC (2018) Estimation of Drug-Target Residence Times by τ-Random Acceleration Molecular Dynamics Simulations. J. Chem. Theor. Comput., 14, 3859–3869.
[4] Kokh DB, Kaufmann T, Kister B, Wade RC (2019). Machine Learning Analysis of τRAMD Trajectories to Decipher Molecular Determinants of Drug-Target Residence Times Front. Mol. Biosci., 6, 36. (P2008)
[5] https://kbbox.h-its.org
[6] Bruce NJ, Ganotra GK, Richter S, Wade RC (2019) KBbox: A Toolbox of Computational Methods for Studying the Kinetics of Molecular Binding J. Chem. Info. Model. 59, 3630-3634. (P2009)
[7] https://github.com/HITS-MCM/MD-IFP
[8] Kokh DB, Doser B, Richter S, Ormersbach F, Cheng X, Wade RC (2020) A workflow for exploring ligand dissociation from a macromolecule: Efficient random acceleration molecular dynamics simulation and interaction fingerprint analysis of ligand trajectories, J. Chem. Phys. 153, 125102. (P2562)

GlyR-GRALL[9], a library/database of data on Glycine receptor allosteric ligands (Cerdan *et al*. 2020[10]), and MoDEL-CNS[11].

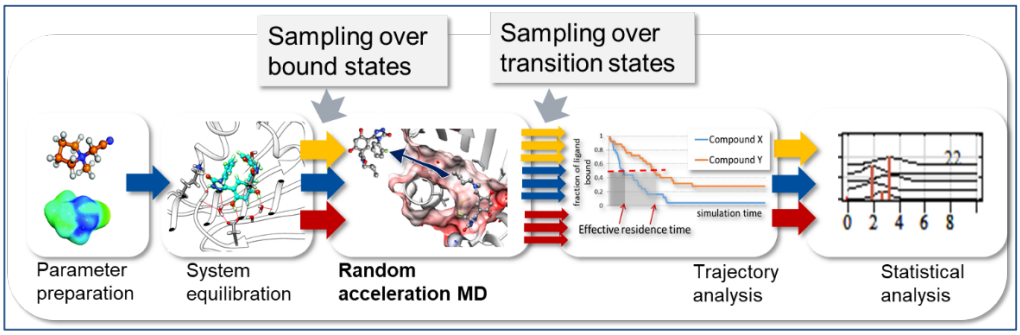


Figure 3: Schematic illustration of the tauRAMD workflow

The tauRAMD workflow computes relative dissociation rate constants for protein-ligand complexes.

3.1.2.1 Outlook

New scripts are being developed to compute dissociation rate constants. We are working on using FENIX resources for the more computationally demanding parts of the scripts and have been allocated ICEI-FENIX resources for the purpose.

Components of the tool set developed in the HBP Collaboratory v1 will be ported to the new EBRAINS platform. A migration example has been generated.

3.1.3 Subcellular web app

Type: web app
Scope: subcellular
Abstraction level: reaction-diffusion systems
Status: Available for use, EBRAINS integration in progress
Homepage & more information:

- https://subcellular.humanbrainproject.eu
- https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/subcellular_level/subcellular_app/subcellular_app.html

The Subcellular web app is a graphical environment for creation and simulation of brain molecular networks. It was designed to reach several objectives related to major limitations of currently available software tools, such as the lack of integration with existing biological data relevant for modelling and the low compatibility of different types of models. It also allows import, combination and simulation of existing models expressed with the BNGL and SBML languages.

Two types of models are supported: rule-based models - convenient and computationally efficient - for modelling big protein signalling complexes and chemical reaction network models.

The app is integrated with a number of solvers for reaction-diffusion systems of equations. It supports simulation of spatially distributed systems using STEPS (stochastic engine for pathway simulation) – which provides spatial stochastic and deterministic solvers for simulation of reactions and diffusion on tetrahedral meshes. It also provides a number of facilities for visualising model geometry and simulation results.

The app is integrated with the Molecular Repository, a publicly available database of biological information, relevant for brain molecular network modelling. It accommodates several types of

[9] https://ifm.chimie.unistra.fr/grall
[10] Cerdan AH, Sisquellas M, Pereira G, Barreto Gomes DE, Changeux JP, Cecchini M (2020) The Glycine Receptor Allosteric Ligands Library (GRALL) Bioinformatics, 36, 3379–3384. (P2442)
[11] http://mmb.irbbarcelona.org/MoDEL-CNS/

biological information which are not available from existing public databases, such as concentrations of proteins in different subcellular compartments of neuronal and glial cells, kinetic data on protein interactions specific for brain and synaptic signalling and plasticity, data on molecules mobility. The Molecular Repository can be queried from the Subcellular app and the results of the query can be added to a molecular network model.

More information can be found in the preprint of Santos *et al.*[12].

3.1.3.1 Recent changes

The application scalability has been improved to support up to 10 million data points.

3.1.3.2 Outlook

The following improvements are planned:

- improved SBML importing
- visualization of reactivity network
- output of spatial solver downloadable
- support additional input format
- complete integration to EBRAINS.

3.1.4 Subcellular model building tool-set

Type: library
Scope: subcellular models
Abstraction level: molecular/subcellular
Status: in progress

We are developing a toolset which can be used in a cohesive workflow, which collects relevant information needed for building and validating models, estimating their parameters, simulating them in different environments and analysing the results. These tools provide a means of bringing in experimental data for modelling in a structured way, clearly defining constraints on model parameters both from previous experiments and molecular simulations and, finally, allow model export for multiple types of end simulations (different simulators). A molecular component will consist of use-cases that guide and demonstrate the use of molecular level information to parameterise and constrain subcellular models (Bruce *et al.* 2019[13]).

The parameter estimation for models of subcellular signalling can be performed through traditional optimisation, using standard libraries, e.g. in MatLab, but also through Bayesian MCMC sampling methods. We are in the process of developing use cases with mcmc_clib/ode_smmala (written in C) (Kramer *et al.* 2014[14]) and also using ABC (written in R) (Eriksson *et al.* 2019[15]) for the sampling of acceptable model parameterisations using a Bayesian approach. These methods allow us to quantify the uncertainty in the model output (propagated from the uncertainty of the parameter estimation

[12] Santos JPG, Pajo K, Trpevski D, Stepaniuk A, Eriksson O, Nair AG, Keller D, Hellgren Kotaleski J, Kramer A (submitted) A Modular Workflow for Model Building, Analysis, and Parameter Estimation in Systems Biology and Neuroscience. https://www.biorxiv.org/content/10.1101/2020.11.17.385203v1. (P2619)
[13] Bruce N, Narzi D, Trpevski D, van Keulen SC, Nair AG, Röthlisberger U, Wade RC, Carloni P, Hellgren Kotaleski J (2019) Regulation of adenylyl cyclase 5 in striatal neurons confers the ability to detect coincident neuromodulatory signals. PLoS Comput. Biol. 15(10). (P2205)
[14] Kramer A, Stathopoulos V, Girolami M, Radde N (2014) mcmc_clib–an advanced MCMC sampling package for ode models. Bioinformatics, 30(20), 2991-2992.
[15] Eriksson O, Jauhiainen A, Maad Sasane S, Kramer A, Nair AG, Sartorius C, Hellgren Kotaleski J (2019) Uncertainty quantification, propagation and characterization by Bayesian analysis combined with global sensitivity analysis applied to dynamical intracellular pathway models. Bioinformatics, 35(2), 284-292. (P1368)

due to, for instance, a lack of data). These software packages will become integrated into the EBRAINS platform in the near future.

More information of the different steps in the workflow is found in the preprint of Santos *et al.*[16].

3.1.4.1 Outlook

An example use case that demonstrates the whole modelling pipeline will be ready within a year: from (i) specifying the subcellular model using a sharable format (SBtab), (ii) to optimising the model parameters (Matlab optimization and/or R/C MCMC), and (iii) to simulate the resulting model in STEPS via SBML (using the subcellular webapp[16]), as well as (iv) integrating the subcellular model as a module in a cellular level neuron model that in turn is simulated using NEURON. Before the end of SGA3, additional use cases will be developed which will generate alternative model parameter sets using the MCMC sampling approaches.

3.1.5 Feature Extraction Tool

Type: web app
Scope: single neurons
Abstraction level: biophysical models
Status: Available now in HBP Collaboratory v1, EBRAINS integration in progress
Homepage & more information:

- https://collab.humanbrainproject.eu/#/collab/1655/nav/66850
- https://kg.ebrains.eu/search/instances/Software/348d66fb-8e82-4e1b-b266-db886d6ca1b9

The Feature Extraction web application, leveraging the eFEL feature extraction library, allows the user to extract a wide variety of electrophysiological features from experimental recordings. Users can work with their own data or make use of contributions by members of the community. With this tool, users can parse the neural activity recorded from individual neurons belonging to different brain regions, or use recordings from neurons simulated via *in silico* experiments. Users may upload their own files, or use pre-recorded traces provided by the community.

The web application has a user-friendly interface that allows filtering of electrophysiological traces according to different criteria (e.g. brain structure, cell type, electrical type, etc.), and makes it possible to visualise and select them in an easy-to-use point-and-click manner (Figure 4).

The output files (in .json format) contain the average values of the extracted features, which can be used for further statistical analysis of electrophysiological behaviour and are suitable for use in individual cell model optimisation through the Hodgkin-Huxley Neuron Builder utility.

[16] Santos JPG, Pajo K, Trpevski D, Stepaniuk A, Eriksson O, Nair AG, Keller D, Hellgren Kotaleski J, Kramer A. A Modular Workflow for Model Building, Analysis, and Parameter Estimation in Systems Biology and Neuroscience (submitted). (P2619)

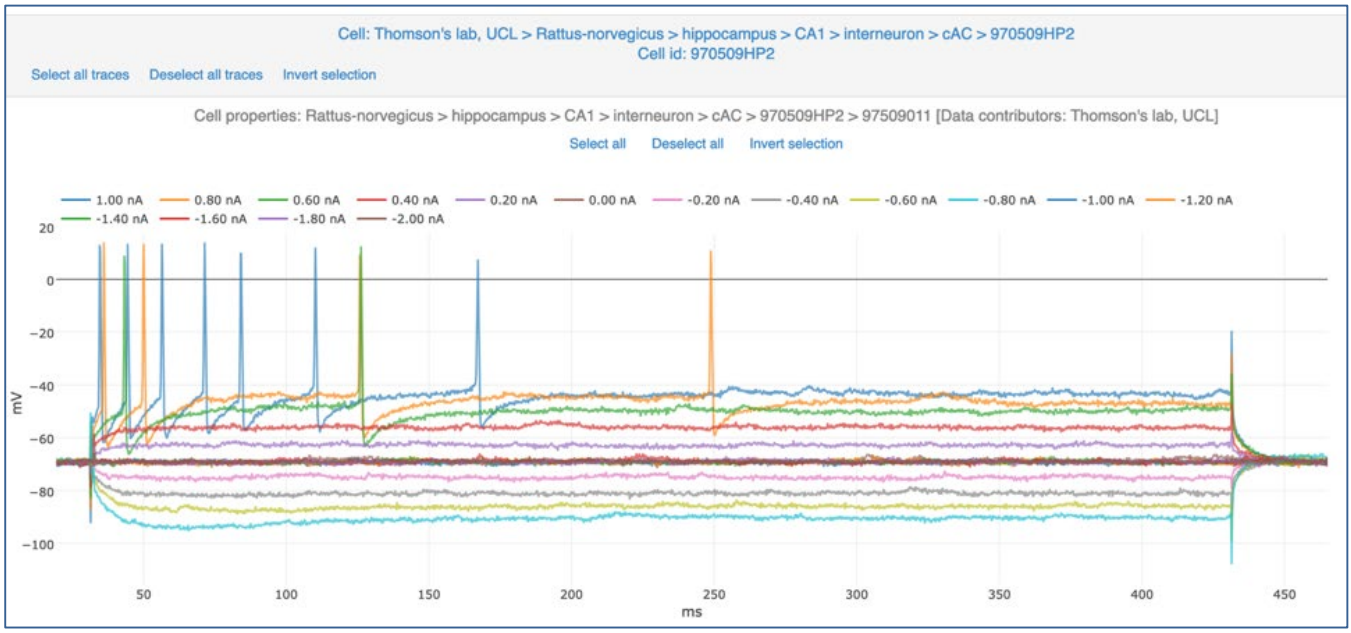


Figure 4: "Feature Extraction" tool interface (trace selection)

3.1.5.1 Recent changes

The tool code has been migrated to the Python 3 programming language.

3.1.5.2 Outlook

Integration with the EBRAINS portal is being finalised. Within the next 12 months, the tool will be provided with functionalities that will allow the users to configure feature extraction parameters (e.g. spike threshold) via the Graphical User Interface. Before the end of SGA3, the integration with the EBRAINS Knowledge Graph (KG) will be finalised (e.g. as for data fetching and provenance tracking).

3.1.6 Hodgkin-Huxley Neuron Builder

Type: web app
Scope: single neurons
Abstraction level: biophysical models
Status: Available now in HBP Collaboratory v1, EBRAINS integration in progress
Homepage & more information

- https://collab.humanbrainproject.eu/#/collab/1655/nav/66852
- https://kg.ebrains.eu/search/instances/Software/0295cfba-f9d6-4743-be2b-4628f7c5ab58

Optimise a single cell model against experimental data on HPC resources using the Hodgkin-Huxley Neuron Builder (Figure 5). With this web-browser application, users can build and optimise their own neuron models, implemented through the NEURON simulation environment, thanks to the BluePyOpt optimisation library. The HPC resources required are offered by EBRAINS free of charge.

For optimisations with the Hodgkin-Huxley Neuron Builder, you may choose a model from the HBP Model Catalog (i.e. the single cell hippocampal models) or upload your own NEURON model files. Enable extraction of the electrophysiological properties used to optimise a NEURON model with the Feature Extraction utility (available on EBRAINS). Once a model has been optimised, it can be fetched from the remote HPC systems and run through the Single Cell *in silico* Experiment tool (BlueNaas) (see documentation) which allows the user to set the simulation parameters and download the simulated activity in text format for further statistical analysis. Both tools are integrated in the web application.

Additionally, the Hodgkin-Huxley Neuron Builder allows the user to save the optimised model in the HBP Model Catalog, which is publicly available to the scientific community.

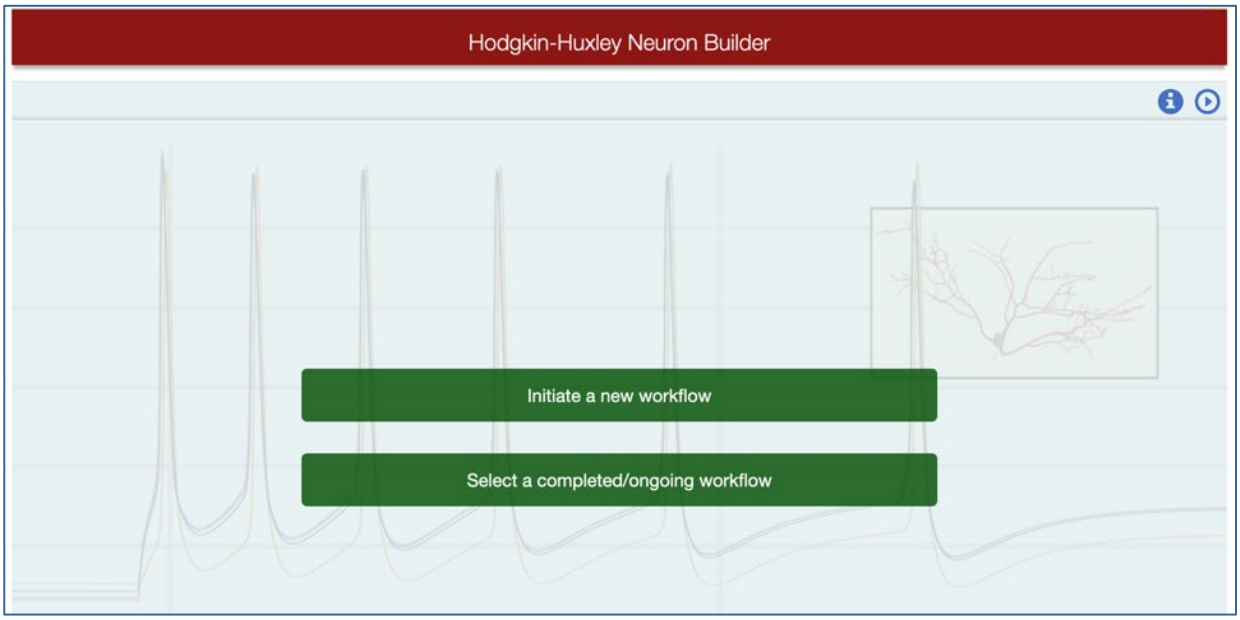


Figure 5: "Hodgkin-Huxley Neuron Builder" tool interface

3.1.6.1 Recent changes

The tool has been improved, and its improvements are being finalised, in view of its integration with the EBRAINS infrastructure (to be implemented in the near future). More specifically, the web application code has been migrated to the Python 3 programming language and the authentication procedures adapted to the new identity and access management system adopted in the EBRAINS infrastructure.

3.1.6.2 Outlook

Full integration with the EBRAINS infrastructure is in progress. Within the next 12 months, the tool will integrate the latest version of the Feature Extraction tool and extend the number of HPC systems available for model optimisation job submission. Before the end of SGA3, the integration with the KG will be finalised (e.g. as for data fetching and provenance tracking).

3.1.7 Electrical model building toolset

Type: command-line tool or library
Scope: single neurons
Abstraction level: biophysical neuron models
Status: mature, available now on HPC, JupyterLab
Homepage & more information:
BluePyOpt:

- https://github.com/BlueBrain/BluePyOpt
- https://kg.ebrains.eu/search/instances/Software/d2918247-8f25-4252-8efb-906e8190d5cd

Feature Extraction Graphical User Interface:

- https://collab.humanbrainproject.eu/#/collab/1655/nav/66850
- https://kg.ebrains.eu/search/instances/Software/348d66fb-8e82-4e1b-b266-db886d6ca1b9

eFEL:

- https://github.com/BlueBrain/eFEL
- https://kg.ebrains.eu/search/instances/Software/42c9ca46-1689-4494-85b5-33f6dffc1ac7

BluePyEfe:

- https://github.com/BlueBrain/BluePyEfe

This toolset consists of 4 tools (eFEL, BluePyEfe, BluePyOpt and BluePyMM) that are used sequentially for building and validating biophysically detailed electrical neuron models. The users start by extracting electrical features from the experimental data using the eFEL library by running the BluePyEfe software on their electrophysiological dataset. Next, they use the extracted features as constraints for a parameter optimisation problem that is solved by the BluePyOpt software. Lastly, BluePyMM is used to validate how well the produced electrical models generalise across different neuron morphologies. These software packages are available as open source Python code to the community and can easily be installed using the Python package installer. They are also integrated in services available via EBRAINS.

Capabilities:

- Extract electrical features from experimental voltage recordings.
- Tune parameters of electrical neuron models to reproduce above mentioned electrical features.
- Generalise these electrical models to previously unseen neuronal morphologies.

3.1.7.1 Recent changes

In the eFEL library a couple of new features were added, the user now also has the option of specifying the precision threshold of certain features. The BluePyEfe package now includes an option to read Axon files. An option was added to BluePyOpt to restart an optimisation for a specified population, enabling users to continue a previous optimisation with other settings.

3.1.7.2 Outlook

For BluePyEfe, support for more standardised formats will be added. BluePyOpt will gain from new optimisation algorithms and integration with LFPy, which is in the works. In eFEL, more and more features will be implemented in Python, instead of C++, to make interactions with users easier. The core algorithm of BluePyMM is being rethought to support very large neuron networks.

In the longer term, we will work on the integration of BluePyOpt with Arbor, the configuration system of BluePyEfe will be reworked to accommodate more general use cases. BluePyMM will support the SONATA network description format.

3.1.8 PyNN

Type: command-line tool or library
Scope: networks
Abstraction level: point neuron models, biophysical neuron models
Status: mature, available now on HPC, JupyterLab, neuromorphic
Homepage & more information:

- http://neuralensemble.org/PyNN/
- https://kg.ebrains.eu/search/instances/Software/bb06ac59-9ada-4b1d-a00a-e2fcafb031f8

PyNN is both (i) a Python API for simulator-independent modelling and simulation of spiking neuronal networks and (ii) a reference implementation of this API for the NEST, NEURON and Brian simulators, distributed as a Python package (library). Implementations of the API are also available for the SpiNNaker and BrainScaleS neuromorphic computing systems; these are developed as separate packages. The principal advantage of PyNN in EBRAINS is that the same code can be used both on software simulators such as NEST and on neuromorphic hardware such as SpiNNaker. PyNN can be installed on HPC systems and in Jupyter notebooks, and is pre-installed on the Neuromorphic Computing platform and the Neurorobotics platform.

3.1.8.1 Recent changes

The latest release of PyNN (v0.9.6) brings support for the most recent versions of NEST and NEURON, and simplifies installation in Windows environments.

3.1.8.2 Outlook

PyNN v0.10 will bring support for the Brian 2 simulator and for the upcoming NEST v3, as well as improved documentation.

Within the next two years, we will release PyNN 1.0, at which point the API for point-neuron models will be considered stable. We will also begin releasing development versions of what will become PyNN 2.0[17], with the major change being support for multi-compartmental neuron models, with implementations for NEURON and Arbor. This will greatly facilitate cross-simulator comparisons for morphologically and biophysically detailed neuronal network models.

3.1.9 NESTML

Type: command-line tool or library
Scope: single neurons, networks
Abstraction level: point neuron models
Status: mature, available now on HPC, JupyterLab
Homepage & more information:

- https://nestml.readthedocs.io
- https://kg.ebrains.eu/search/instances/Software/95472c51-772c-42dd-a427-553851f83c2b
-

NESTML is a domain-specific language that supports the specification of neuron models in a precise and concise syntax, based on the Python syntax. Model equations can be written as a simple string in mathematical notation, and as an algorithm using the built-in procedural language. Differential equations are analysed by the associated toolchain, written in Python, to compute an exact solution, if possible, or otherwise use an appropriate numeric solver. Code may then be generated for a target platform such as NEST Simulator (in place) or SpiNNaker (SGA3 goal). This code is typically in a lower-level language such as C++ or assembly and optimised for simulating particular neuron and synapse models on that particular hardware.

3.1.9.1 Recent changes

An architectural design has been formalised that extends NESTML with the capacity to cover third-factor plasticity rules in a flexible and open-ended manner. This will allow all third-factor synaptic plasticity models currently in NEST to be written as NESTML models rather than as C++ files. Requirements for the new language mechanisms were formalised in close communication with end users.

NESTML code generation has been extended to replace more parts of legacy (and inflexible) NEST Simulator C++ code, especially those related to synaptic plasticity.

A prototype has been made for extending the range of applicability of ODE-toolbox, a software component used internally by NESTML. The improvements enable the automatic detection of conditions that would otherwise cause the simulation to fail, and select an alternative integration method accordingly.

The documentation and training materials have been improved, making use of our experience presenting interactive NESTML workshops and tutorials.

3.1.9.2 Outlook

- Next major release of NESTML (version 4.0) containing improvements to equations and kernels handling.
- Next minor release of ODE-toolbox (3.1) containing bug fixes.

[17] http://neuralensemble.org/docs/PyNN/2.0/

- Release of alpha-stage NESTML prototype for co-generation of neuron and synapse models. This is a fully functional implementation of the new NESTML architecture that will support the efficient generation of synaptic plasticity code. Its release will allow end-users to play with the new functionality and provide continuous, valuable feedback based on actual end-user requirements. In the meanwhile, we will continue to work internally on a fully transparent integration with NEST Simulator, using the Just-In-Time code generation architecture presented at NEST Conference 2020.

Together with our collaborators in the SpiNNaker team, we will formulate a roadmap for the NESTML SpiNNaker target code generation, which will be fully functional, and at a high TRL, at the end of SGA3.

3.1.10 Snudda

Type: command-line tool or library
Scope: networks
Abstraction level: biophysical neurons
Status: prototypes available
Homepage & more information: https://github.com/Hjorthmedh/Snudda

Snudda is an open-source tool which creates networks of detailed neurons, where the connectivity of the network is based on the morphologies of the neurons. It simulates network models written in NEURON and NMODL, and stores related data in JSON and HDF5 formats. To integrate the created networks into the pipeline, we will develop the SONATA export feature. Snudda is specialised on subcortical non-layered structures, and incorporates dynamic neuron modulation. The programme can also be downloaded and run locally on a private workstation, or installed on a supercomputer for generation and simulation of larger networks. Morphological reconstructions are curated, scaled and resampled to ensure the standardised appearance and digital representation in a semi-automatic procedure using assistant Python scripts. Morphologies cut at the slice border are attempted to get repaired. Repaired and normalised morphologies are further validated via HBP Morphology Validation Framework.

3.1.10.1 Recent changes

Since publication of Hjorth *et al*. 2020[18] the codebase has been refactored and reworked to conform with the PEP standard. Input specification and generation for the network has been updated to give more flexibility in controlling what individual neurons and populations of neurons receive as input. Morphologies have been standardised to 3-micrometre resolution throughout and new morphologies have been added to the library of available neurons.

3.1.10.2 Outlook

A new, more general framework for setting up neuromodulation in the network is being developed. The input dynamics for the synapses are being optimised to account for new data on the NMDA/AMPA currents. Tools are being designed to allow for specifying population units, and to introduce ablation of connectivity for virtual experiments.

Updated SONATA support and a tighter integration with the HBP workflow is planned for the future. We plan to continuously update the network model, including adding new morphologies and neuron models.

[18] Hjorth JJJ, Kozlov A, Carannante I, Frost Nylén J, Lindroos R, Johansson Y, Tokarska A, Dorst MC, Suryanarayana SM, Silberberg G, Hellgren Kotaleski J, Grillner S (2020) The microcircuits of striatum in silico. Proc. Natl. Acad. Sci. 117 (17) 9554-9565. (P2489)

3.1.11 Brain Scaffold Builder

Type: command-line tool or library
Scope: networks

Abstraction level: multi -scale networks with spatial features, biophysical neurons, point neurons
Status: prototypes available
Homepage & more information:
- https://github.com/dbbs-lab/bsb

A modelling framework with an all-encompassing set of multi-scale features with special attention to biological realism: from describing complex high-level topologies such as brain regions, cortical structures or local microcircuitry, down to specifying parameters on single synapses and customising connectivity strategies to target subcellular spatial features of the cell morphologies. Topological regions can be defined and cells can be placed and connected using built-in and user-defined placement and connectivity strategies. The design of the framework and how it separates model description from execution allows for rapid iterations and modifications to the description of the model and to recreate smaller modified parts of the model. Although the framework aims to make next-level biological realism in network reconstruction achievable, it also offers managed parallel simulations on multiple simulator back ends (NEST, NEURON and - soon - Arbor). The BSB framework comes with several useful modules such as a CLI, a plotting and even 3D-animation module. It is freely available as open-source code, distributed as a Python package on PyPI and up to standards with testing, test coverage and documentation using industry standard CI/CD and development practices.

3.1.11.1 Recent changes

Since the start of SGA3, we have added Blender support, a new morphology module and the powerful new configuration module, which will be available to the public in the next major release 4.0. We have tested the framework for the cerebellar cortex, a challenging brain region to model spatially with anisotropic morphologies and highly specific connectivity rules, the results of which will be submitted soon.

3.1.11.2 Outlook

A major new version is coming soon with improved design and public interfaces, support for Arbor and LFP simulations and more support for community supported data formats for both network description and simulation results.

Long-term plans include optimising our algorithms, improving the public API, based on user feedback, and adding more placement, connectivity and post-processing options.

3.1.12 The Virtual Brain (TVB)

Type: command-line tool or library & web app
Scope: whole brain

Abstraction level: population models
Status: mature, available, EBRAINS integration well advanced
Homepage & more information:
- https://www.thevirtualbrain.org/tvb/zwei
- https://kg.ebrains.eu/search/instances/Software/316edfdb-abd7-4451-b54e-21b746838ec0

The Virtual Brain (TVB) is an open-source platform for constructing and simulating personalised brain network models. The TVB-on-EBRAINS ecosystem includes a variety of pre-packaged modules, integrated simulation tools, pipelines and data sets for easy and immediate use on EBRAINS. Users can process their large cohort databases and use these results to develop potential medical treatments, therapies or diagnostic procedures.

3.1.12.1 Recent changes

The current code available in the master branch of the repository already includes RateML, a domain specific language developed during SGA2 and which is being refined during SGA3. RateML allows scientists to define new models for TVB, automatically generating code in Python and CUDA. The CUDA backend allows users to deploy large parameter fitting workflows on the ICEI infrastructure in an easy and well-integrated manner. Examples for this are now available in the Collaboratory v2 using PyUnicore: https://wiki.ebrains.eu/bin/view/Collabs/rateml-tvb/.

We have tested the current TVB simulation infrastructure, especially the Collaboratory v2, including porting IPython notebooks, compiling required libraries for Bayesian inference and writing an accelerated CPU version of models required to support scientific workflows for high-performance parameter fitting and personalised modelling.

Substantial effort has also been devoted to the implementation of TVB-NEST co-simulation proofs of concept, in collaboration with the multi-simulator interaction tasks.

We have had two workshops to show these new capabilities to TVB users within the HBP and a poster in the Bernstein Conference to invite the community to try new features.

Our new users include researchers with new models which need to be added to TVB in a simple way, which also directly integrates to other EBRAINS components as access to data and large parameter space explorations. With the integration of Bayesian inference workflows, we also invite users in clinical research to explore the high potential of model personalisation through EBRAINS.

We are developing, refining, testing and integrating a set of software tools and interfaces to enable the cohesive operation of TVB with EBRAINS workflows. The main software components and interfaces targeted include the compatibility with data formats used to interact with the Brain Atlas and the KG, the communication modules between simulators, Collaboratory, Neurorobotics platform, Human Intracerebral EEG Platform (HIP), and the development of a testing suite for workflows containing TVB simulations.

In the last 9 months, we focused on enabling the connection between data and models through automatic code generation and on the communication between TVB and other EBRAINS components.

3.1.12.2 Outlook

RateML is being extended to allow scientists to define links between data coming from the KG and model variables. This feature is of great interest, given that scientists would like to dynamically explore the relationship between different multimodal data features in their models in a robust and reproducible manner. This code is still in development in a repository (https://github.com/DeLaVlag/tvb-root.git) and will be merged with the master branch as soon as it has been tested and is ready.

In the next two years, we will focus on testing, validation, provenance tracking, deployment and integration of complete workflows which link TVB with multiple EBRAINS tools and allow the end users to fully leverage the easy access to HPC, large data sources and the shared knowledge that EBRAINS provides.

3.2 Running Simulations

3.2.1 CGMD Platform

Type: web app
Scope: subcellular

Abstraction level: coarse-grained molecular dynamics
Status: available for use, EBRAINS integration in progress
Homepage & more information:

- https://molsim.sci.univr.it/mermaid/begin.php

Integrated web-servers for the preparation, run and analysis of coarse-grained molecular dynamics simulations. The server allows the simulation of membrane and soluble proteins using the Martini and the SIRAH force fields, as well as an automated protocol for carrying out the back-mapping of the coarse-grained description of the system into an atomistic one. Figure 6 shows a screenshot of the CGMD platform.

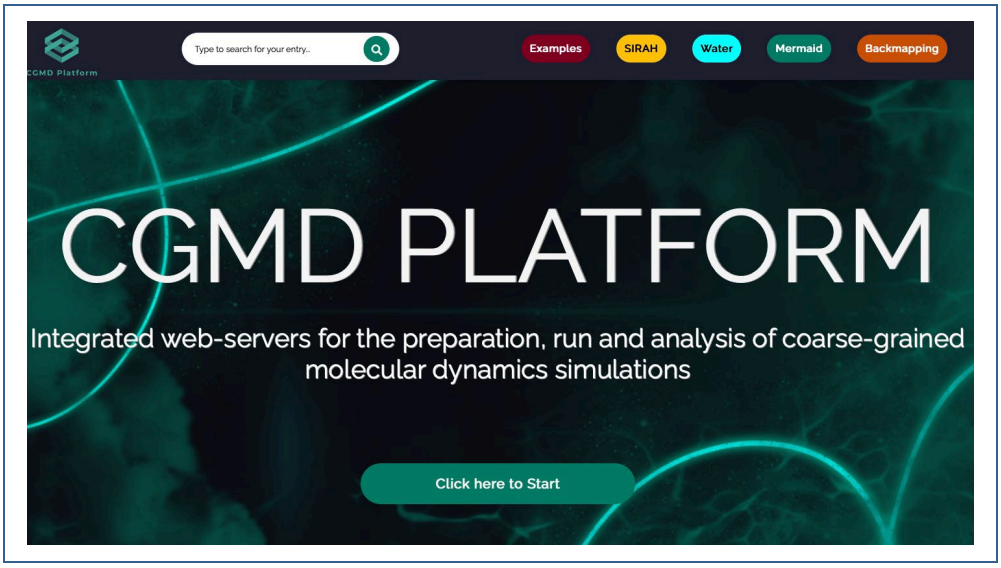


Figure 6: Screenshot of the simulation platform

3.2.1.1 Recent changes

The webserver is online, working and already available via the Collaboratory v2[19].

3.2.1.2 Outlook

The next steps will include the embedding of the server within the EBRAINS platform and more features will be added in the longer term.

3.2.2 SSB Platform / METNET

Type: web app and standalone app
Scope: subcellular
Abstraction level: Combined molecular and systems biology simulation
Status: In development

Quantitative Kinetic Metabolic Networks Modelling. Predict the downstream effects upon ligand binding/disease associated variants such as (protein/metabolites concentrations/dose response curves). The application will allow users to explore metabolic networks using automated bio-mathematical models, by integrating a molecular detail description of the protein/ligand interaction with a systems biology simulation of the signalling pathway. Figure 7 shows the initial page of the MetNet protocol.

[19] https://wiki.ebrains.eu/bin/view/Collabs/molecular-tools-cgmd-platform/

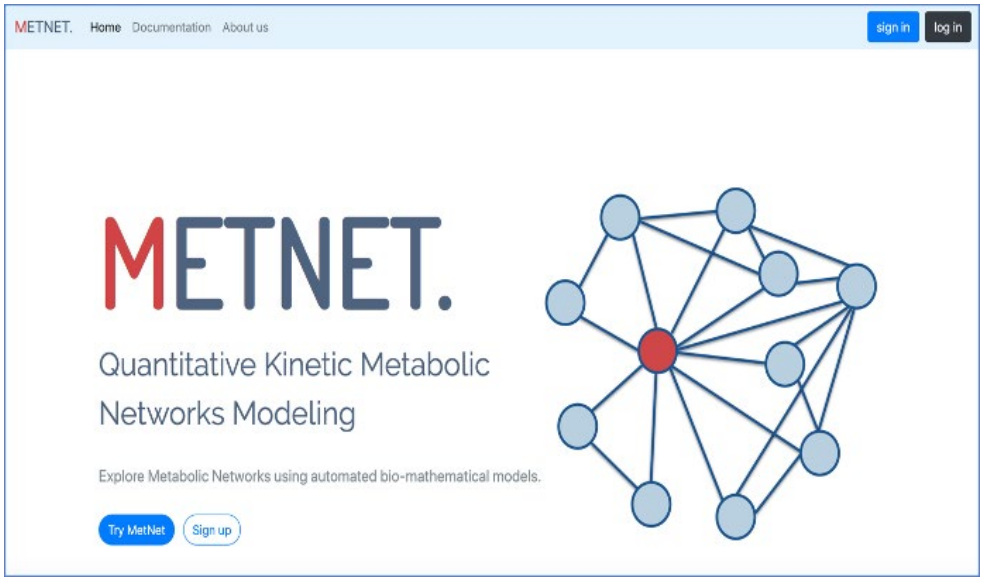


Figure 7: Initial page of the MetNet protocol

3.2.2.1 Outlook

We are finalising the realisation of a set of stand-alone Jupyter notebooks for the simulation protocol. This will be deposited and shared within the EBRAINS platform. In the longer term, we plan to develop a webserver to be implemented in the EBRAINS platform.

3.2.3 MoDEL-CNS

Type: web app
Scope: subcellular
Abstraction level: Atomistic Molecular Dynamics
Status: Beta version publicly available on-line and within the EBRAINS infrastructure
Homepage & more information:

- http://mmb.irbbarcelona.org/MoDEL-CNS/
- https://wiki.ebrains.eu/bin/view/Collabs/molecular-tools-model-cns/

MoDEL-CNS is a web server connected to a BigData (MongoDB) database giving access to atomistic Molecular Dynamics trajectories for relevant signal transduction proteins. The server shows in a graphical way a set of flexibility analyses pre-computed on the trajectories. A REST API makes it possible to download programmatically all data including trajectories and analyses. Users interested in protein flexibility can easily identify different protein conformations, flexible regions or interesting flexibility descriptors using the graphical interface. Frames containing this information can then be easily downloaded for new studies (e.g. Docking) using the REST API. Figure 8 shows a screenshot of the MoDEL-CNS Web Server in the Collaboratory v2.

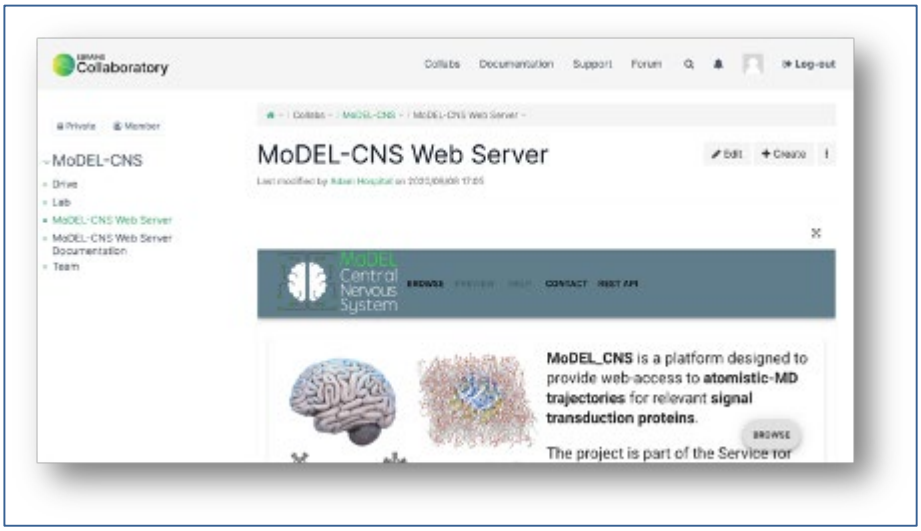


Figure 8: Extract of the MoDEL-CNS web server integrated in the EBRAINS platform

3.2.3.1 Recent changes

Integrated in the EBRAINS platform: https://wiki.ebrains.eu/bin/view/Collabs/molecular-tools-model-cns/.

3.2.3.2 Outlook

The collection of MD trajectories and flexibility analysis is being extended thanks to the use of ICEI/FENIX resources (CSCS Piz Daint supercomputer). The available set of analyses is going to be extended as well, adding new studies protein family-specific.

Jupyter Notebooks retrieving information from the MoDEL-CNS REST API will be developed and integrated in the EBRAINS platform using the integrated Jupyter Lab tool.

3.2.4 SDA

Type: web app and standalone app
Scope: molecular
Abstraction level: nanoscale
Status: Available for use
Homepage & more information:

- https://mcm.h-its.org/sda
- https://websda.h-its.org/

SDA (Simulation of Diffusional Association) is a Brownian dynamics simulation code that can be used to simulate the diffusion of biomacromolecules in aqueous solution. It is particularly useful for studying the effects of electrostatic steering on molecular diffusional association processes. The association of two molecules can be simulated to compute bimolecular diffusional association rate constants under dilute conditions and to predict the structures of diffusional encounter complexes, subject to biochemical constraints. SDA can also be used to simulate concentrated protein solutions and to investigate the diffusional association of proteins with solid surfaces. SDA 7 is available for standalone use and some of the functionality is implemented in the webSDA webserver. Figure 9 shows an illustration of the SDA applications.

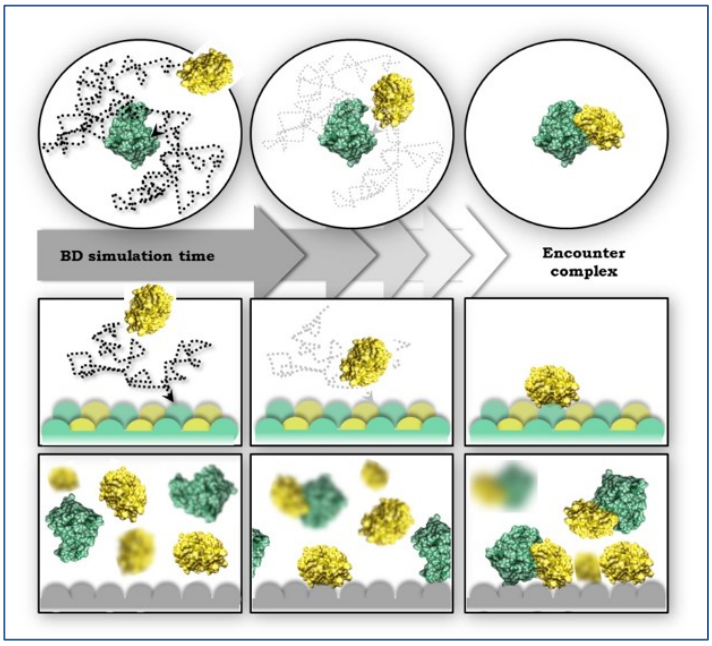


Figure 9: Illustration of the application of SDA

Illustration of the application of SDA for simulating the diffusional association of two proteins (top), the adsorption of a protein on a surface (middle), and many proteins in the presence of a surface (bottom).

3.2.4.1 Recent changes

SDA 7.3.0, which was released in December 2020, has added functionality for the modelling of the interactions of proteins with surfaces allowing inclusion of protein-surface hydrodynamic interactions and long range electrostatic interactions.

3.2.4.2 Outlook

Planned developments are aimed at combining molecular dynamics and Brownian dynamics simulations to compute association rate constants for protein-ligand binding accounting for molecular flexibility.

3.2.5 Single Cell In Silico Experiment Tool (BlueNaaS)

Type: web app
Scope: single neurons
Abstraction level: biophysical models
Status: Available now at Collab v1 BSP. EBRAINS integration done
Homepage & more information:

- https://blue-naas-bsp-epfl.apps.hbp.eu/#/
- https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/single_cell_in_silico/single_cell_clamp/single_cell_clamp.html

BlueNaaS ("Neuron as a Service") is a web application that simulates one single neuron model in the web browser. It supports the use case “Single cell *in silico* experiments under current/voltage clamp” and several Live Papers.

3.2.5.1 Recent changes

- Migration from legacy system to EBRAINS.
- Support for Live Papers (API modifications).

3.2.5.2 Outlook

The application will be maintained, but no more features will be added.

3.2.6 Small Circuit In Silico Experiments Tool (Rat Hippocampus CA1)

Type: web app
Scope: networks
Abstraction level: biophysical neurons
Status: Available, EBRAINS integration almost done
Homepage & more information:

- https://bsp-rat-ca1.humanbrainproject.eu/circuits/rat-ca1
- https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/small_circuit_in_silico_experiments/pair_recording_app/pair_recording_app.html

This web application allows users to select individual cells from a large-scale detailed network model to build and run a small hippocampal neural circuit (based on the Ascoli atlas). Figure 10 shows the tool interface.

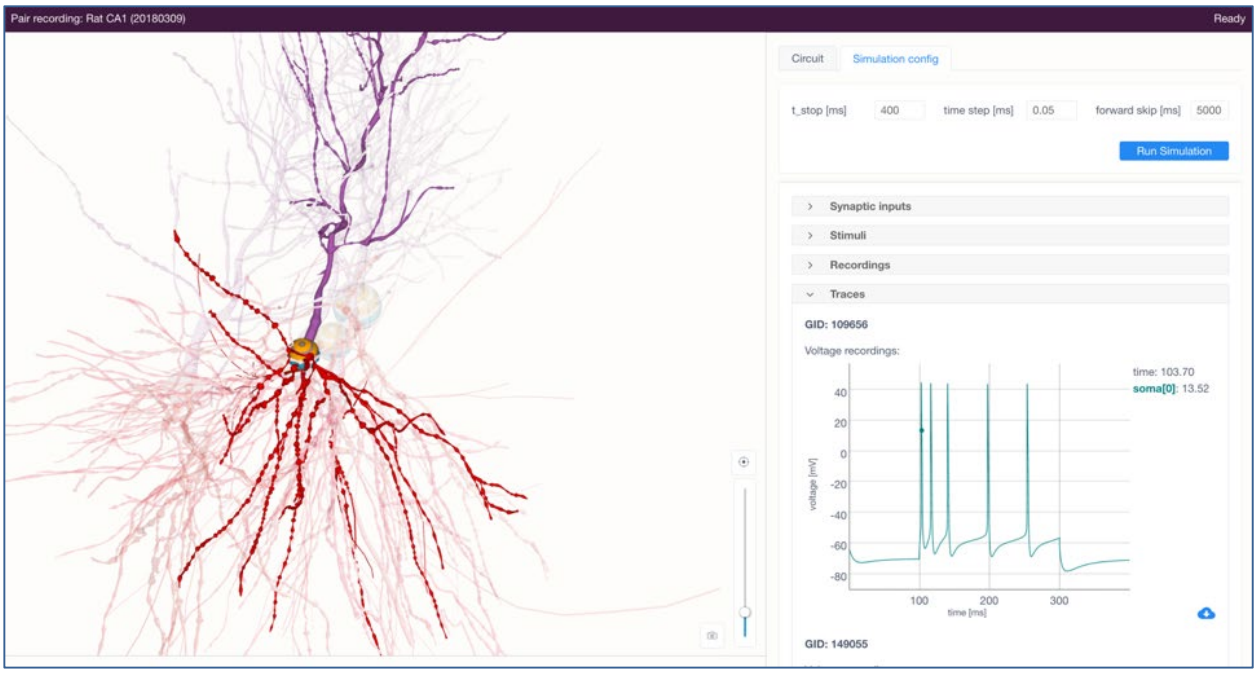


Figure 10: "Small Circuit *In Silico* Experiments" tool interface

3.2.6.1 Recent changes

The web application has been improved with additional functionalities: the possibility to run longer simulations (up to 30 s) has been added, as well as the support for the SONATA format, in view of a novel MOOC (currently being finalised).

The deployment of the application is two-fold. The tool has been deployed on a dedicated site, in an EBRAINS-compatible manner (i.e. leveraging the services of the EBRAINS infrastructure, such as the EBRAINS authentication system). At the same time, the tool is still available to the community in the HBP Collaboratory v1 environment.

3.2.6.2 Outlook

EBRAINS integration will be completed in the near future.

3.2.7 *Brain Areas Circuit In Silico Experiments Tool (Rat Hippocampus CA1)*

Type: web app
Scope: networks
Abstraction level: biophysical neurons
Status: available, EBRAINS integration done
Homepage & more information:

- https://simulation-launcher-bsp-epfl.apps.hbp.eu/index.html#/circuits/hippo_hbp_sa_full_ca1
- https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/brain_area_circuit_in_silico_experiments/hippocampus/configure_run_brainregion_preconf_model_data/configure_run_brainregion_preconf_model_data.html

Large network simulation for very large networks of detailed neuron models. It supports the use case "Configure and run a rat hippocampus CA1 region using preconfigured HBP model and data" and a MOOC for *in silico* experiments for the hippocampus model. Figure 11 shows the tool interface.

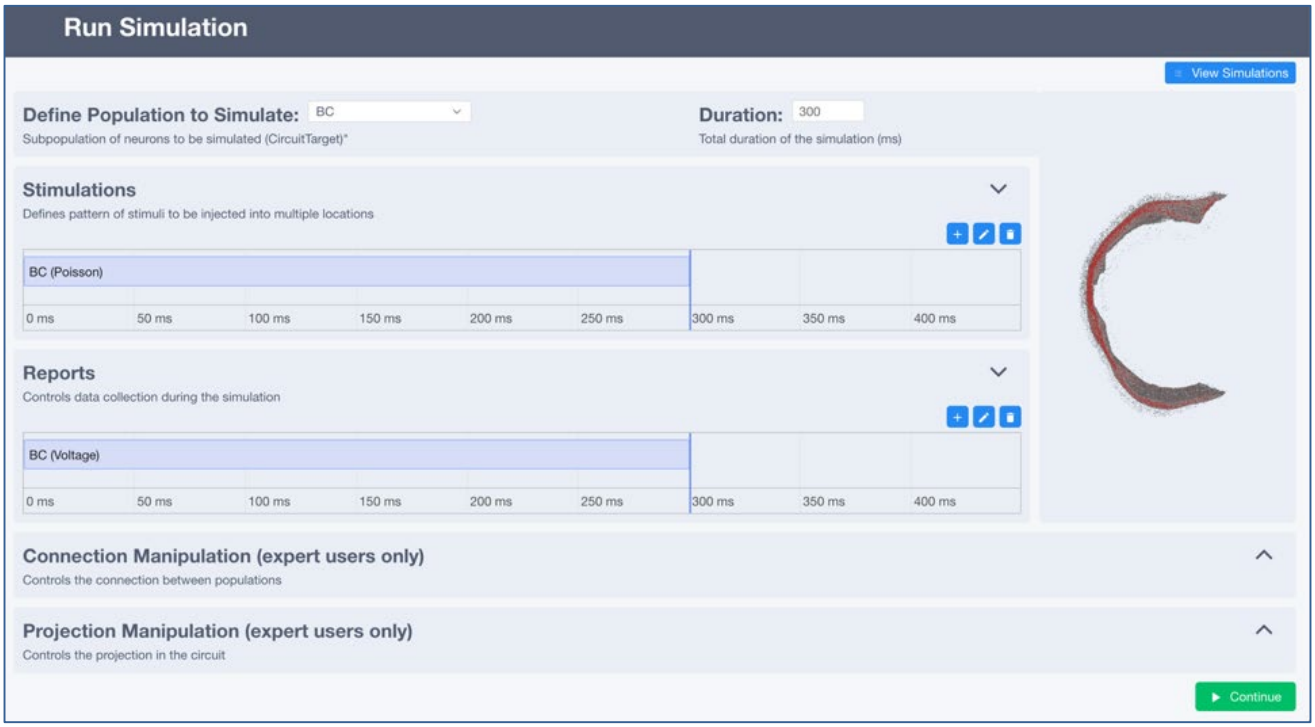


Figure 11: "Brain Area Circuit" tool interface

When required, the tools access the FENIX infrastructure to leverage its HPC resources; currently available at CSCS (Lugano, Switzerland), JSC (Juelich, Germany), CINECA (Bologna, Italy) and NSG (San Diego, USA). When HPC resources are needed, the users exploit them using their credentials on the HPC systems (via accredited PRACE grants). Users with no HPC accreditation can still run simulations, thanks to a dedicated service account functionality integrated in the relevant tools.

3.2.7.1 Recent changes

In view of an upcoming new MOOC currently being finalised, this tool has been improved with support for the SONATA format, as well as with new visualisation utilities. Additionally, this application has been extended so it can be run with dynamically generated neural circuits. The tool is currently deployed in both the EBRAINS and the HBP Collaboratory v1 environments.

3.2.7.2 Outlook

The tool will be maintained and supported going forward.

3.2.8 NEST

Type: command-line tool or library
Scope: networks
Abstraction level: point neuron models
Status: mature, available on HPC and JupyterLab
Homepage & more information:

- https://nest-simulator.org
- https://kg.ebrains.eu/search/instances/Software/c3d23fd5-4370-487b-9231-e0f8e8a1e9e6

NEST is a simulator for spiking neuronal networks, based on over 20 years of development. The tool has an active developer and user community that continuously provides neuroscientists with updates and improvements to all aspects of the technology. NEST thereby remains relevant to current research trends. NEST is scalable and allows researchers to develop plastic networks of point-neuron models of natural size and density for simulations with any computing system that fits their needs. The development team tests and verifies the code to ensure accuracy, and ensures compatibility and interoperability with other modelling tools like PyNN. The user interface is Python-based, the scripting language that the computational neuroscience community has agreed upon. This helps neuroscientists to immediately use NEST productively and facilitates the interfacing with other tools used by the neuroscience community. NEST can be installed on several types of platform, with continuous developments to improve on-boarding via simplified installation procedures using virtual machines, Conda and Docker. The technology of NEST is well documented by a large number of peer-reviewed technical papers. The reliability and usefulness is also proven by the large number of peer-reviewed neuroscience publications produced with NEST. Furthermore, NEST has become a standard for the verification and validation of neuromorphic hardware systems. The scope ranges from educational simulations on laptops to brain-scale simulation on world-leading supercomputers, and from computational neuroscience through validation and benchmarking of neuromorphic hardware to neurorobotics.

3.2.8.1 Recent changes

The NEST Simulator was extended and now offers the possibility of using third-factor plasticity in models. Function has been demonstrated on the examples of Clopath plasticity and the Urbanzcyk & Senn plasticity rule, which are also described in the user-level documentation of NEST.

Dedicated NEST developer hackathons focused on code revision and specific developments to ensure continued long-term sustainability of the code base. At the same time, the code sustainability was facilitated by actively strengthening the NEST community. The very successful online-only NEST Conference 2020[20] attracted a larger and much more international audience than previous on-site events, and brought developers and users together. Users, developers and experts worked together in hackathons focussing on specific features. Aside from the conference and hackathons, NEST developers meet every other Monday for an Open Developer Video Conference to discuss current topics in development and review open pull requests, issues and mailing list requests. Between 25 and 40 developers have contributed code, documentation or examples to recent NEST releases. Systematic code review, continuous integration testing and refactoring maintain and improve NEST code quality, ensuring long-term sustainability.

New developments in the inner workings of the simulator removed performance bottlenecks during network creation. Large hierarchical networks can now be created faster and make full use of available parallel resources to instantiate the structures in memory.

Latest developments include a REST-ful interface that can be used for co-simulation, visualisation and links to neurorobotics. Most prominent is the classroom teaching-application NEST Desktop.

[20] https://indico-jsc.fz-juelich.de/event/115/

With these advances, it becomes possible to run simulations with a wider range of neuron and synapse model types, express network models in code with higher performance and provide point neuron spiking network simulations to many other parts of EBRAINS.

No other spiking network simulator with point-neuron resolution provides such versatile interoperability with other tools or allows models to grow seamlessly with the user's experience, from laptops to the largest supercomputers.

3.2.8.2 Outlook

In the coming months, work will commence on establishing HPC back-end access from front-end services like Jupyter and NEST Desktop though EBRAINS middleware services. This enables users to exploit EBRAINS simulation capabilities without installation and more seamlessly with other EBRAINS integrated tools and services. The efforts will be supported by the integration of documentation with that of other EBRAINS tools, as jointly started at the "Winter of Documentation" kick off during the CodeJam#11. In parallel, developers will continue maintenance and to support changes introduced into code base according to user needs, as for example supported by different joint voucher projects.

For the mid-term, developers are working towards a higher integration of NESTML-generated model code to replace code that is currently manually curated. This will enhance maintainability of the code base and lower the threshold for modellers. Work is ongoing to establish interoperability within the co-simulation framework and the full integration into the EBRAINS scientific workflow tool chains (Figure 1). As in the short term, developers continue to support users' work through all the support channels (mailing list, HLST, fortnightly VCs, Issue tracking, etc.), regular meetings at workshops and major conferences, and the yearly NEST Conference.

3.2.9 Arbor

Type: command-line tool or library
Scope: single neurons, networks
Abstraction level: biophysical models
Status: prototype available on HPC
Homepage & more information:

- https://arbor.readthedocs.io
- https://kg.ebrains.eu/search/instances/Software/80d205a9-ffb9-4afe-90b8-2f12819950ec

Arbor is a portable, high-performance library for computational neuroscience simulations with multi-compartment, morphologically-detailed cells, ranging from single cell models to very large networks. It is written from the ground up with many-CPU and GPU architectures in mind and can be used with contemporary and future HPC systems to meet simulation needs effectively.

Arbor's performance portability is due to back end-specific optimisations for CPUs from Intel (AVX, AVX2 and AVX512) and ARM (Neon and SVE), as well as NVIDIA and AMD GPUs. When coupled with low-memory overheads, these optimisations make Arbor an order of magnitude faster than the most widely-used comparable simulation software.

Arbor can be downloaded as a C++ library and integrated in your own programme, or installed as a Python library (through pip) and imported in any Python scripts.

Features:

- Run morphologically detailed simulations
- Python and C++ API, for excellent performance at any computational scale
- Load morphologies (SWC, NeuroML) and dynamics (NMODL)
- Easy-to-use domain-specific language (DSL) to describe regions and locations
- Access cellular mechanism catalogues, including mechanisms used by Allen and BBP models
- Run single cell models from databases such as the Allen Brain Atlas

3.2.9.1 Recent changes

Recent developments include: significantly improved documentation; addition of a powerful, unparalleled morphological domain-specific language; full features available via the Python interface; 7 new collaborations with new partners (already started or in planning); and an increased presence at conferences (CNS2020, Bernstein 2020).

3.2.9.2 Outlook

Two major collaborations, FIPPA and ArborIO, were started in October 2020 via Calls for Expressions of Interest) and will continue until the end of SGA3. The CEoIs are spurring development targeted at building novel and ambitious large network models. A smaller project to port of an Openworm simulation has started, with initial discussions to increase the scope of the port.

In the short to medium term, the following features are expected to be delivered:

- Arbor Graphical User Interface, to help to on-board new users to Arbor and help visualise complex models
- Dynamic loading of user-supplied cell dynamics
- Spike-timing dependent plasticity (FIPPA)
- Simulations with gap junction connections distributed over a network (ArborIO)
- Support for efficient calculation of LFPs, for integration with LFPy
- New and expanded file format supports for models and mechanisms, including NeuroML dynamics and Neurolucida morphologies
- EBRAINS integration (availability in the lab, Docker image, preparation of more notebooks)

In the longer term, further integration into the EBRAINS infrastructure and ecosystem of tools is planned. Concretely, integration with optimisers (such as BluePyOpt) is planned, along with support for SONATA model descriptions and co-simulation with NEST.

3.2.10 NEURON/CoreNEURON

Type: command-line tool or library
Scope: single neurons, networks
Abstraction level: biophysical
Status: stable, available on HPC and JupyterLab
Homepage & more information:

- https://github.com/BlueBrain/CoreNeuron
- https://github.com/neuronsimulator/nrn
- https://kg.ebrains.eu/search/instances/Software/dbe2802c-ebe5-4963-a02d-508e07c4be31

NEURON is already well established as a community simulator for networks of detailed neurons. CoreNEURON is developed as a new simulation engine executed within the NEURON ecosystem, which seeks to optimise simulation capacity and performance on next-generation hardware.

3.2.10.1 Recent changes

Recent updates to CoreNEURON include support for more fine-grained reporting, so users can report on a select range of sections. Previously, the user had only the choices of reporting the soma or all compartments. With this update, users are able to select, for example, a few axonal sections to report. For large simulations, this can greatly reduce the amount of IO performed and improve efficiency, since users are not burdened with reporting additional data that do not interest them. Also, with regards to reporting, reports now can use the SONATA specification for output. This includes spike reports as well as compartment reports.

CoreNEURON now supports SEClamp type stimulus from NEURON, so that users have that additional mechanism to use in simulation. Work has also been done to assist users in adapting synapse models to use runtime constructs to change the weight of connection strengths during simulation runtime.

In addition, attention has been paid to general maintenance for both NEURON and CoreNEURON, to improve build options and expand the tests done for CI/CD. NEURON, for example, has the capacity to build using cmake now, rather than only the legacy autotools.

Those interested in the NEURON Dev community have been having regular monthly meetings since March 2020. Notes from the previous meetings can be found on the NEURON GitHub wiki space[21].

Work has also been done to better support GPU integration in NEURON and porting of the olfactory 3D bulb model to run on CINECA's Marconi100 system[22].

3.2.10.2 Outlook

For CoreNEURON, work is under way to replace MOD2C and make NMODL Framework the default code generation tool for MOD file to C++ translation. By the end of SGA3, NEURON should also be adapted to use NMODL Framework, rather than its original translation tool.

In addition, work will be invested in GPU optimisation and support for AMP/Intel GPUs.

Lastly, a validation framework for mod file optimisation will be deployed that will attempt to make qualitative comparisons for when a mechanism uses different optimisations that can lead to different evaluations.

3.2.11 Neuromorphic Computing Job Manager app

Type: web app, command-line app and library
Scope: networks
Abstraction level: point neurons
Status: stable, available in HBP Collaboratory v1, EBRAINS integration in progress
Homepage & more information:

- https://electronicvisions.github.io/hbp-sp9-guidebook/using_the_platform.html

Network models described using the PyNN API can be simulated on the BrainScaleS and SpiNNaker platforms in batch mode by submitting jobs to a central job queue. Three different clients - a web app, a Python client, and a command-line application - are available to anyone with an EBRAINS account. These tools can all be used both to submit jobs and to view/download results. The app is currently available in the HBP Collaboratory v1 and the Python client, which includes the command-line application, can be downloaded from the Python Package Index.

3.2.11.1 Outlook

In the near future, it will become possible to use the Python client from within Jupyter Notebooks in the new Collaboratory v2. Following that, the web app will be ported to Collaboratory v2. Finally, all job records (including provenance metadata) will be migrated to the KG, facilitating workflows that combine HPC and neuromorphic computing.

[21] https://github.com/neuronsimulator/nrn/wiki
[22] https://github.com/HumanBrainProject/olfactory-bulb-3d

3.2.12 Multiscale co-simulation framework

Type: command-line tool or library
Scope: networks, whole brain
Abstraction level: point models, biophysical models, population models
Status: in progress
Homepage & more information:
- https://github.com/multiscale-cosim

The multiscale co-simulation framework includes tools, workflows, protocols and interfaces enabling runtime exchange of neuroscientific information between simulators at different scales of abstraction (Arbor, NEST, TVB, and Gazebo), analysis tools (Elephant) and visualization engines (Insite, Viola, NEST Desktop) (Figure 12). The framework components are designed for High Performance Computing and include the necessary transducing and translation modules between the different scaled of abstraction. The multiscale co-simulation framework allows the investigation of novel questions at the interface between scales by enabling the coupling of large-scale whole brain simulations capturing global brain behaviour with detailed spiking or morphological detailed networks.

The co-simulation framework has three types of end-user: scientists; Developers of neuroscience simulators or tools; and operators responsible for day-to-day operations of the framework.

For scientists, the current phase saw:

1) The first version of the LFPykit which allows the generation of LFP signals from simulations running in the Arbor simulator.
2) Two NEST/TVB co-simulation workflows were implemented specialising on single node end-user flexibility and large scale HPC deployment respectively.
3) Major upgrades in the usability of the NEST Desktop application.

For developers of neuroscience simulators or tools:

1) The NEST server is now part of the standard NEST code-base enabling coupling to NEST Desktop, the Neuro Robotics Platform, in-transit visualisation (Viola, Insite), and Elephant.
2) HPC ready data exchange tools have been integrated in a TVB-NEST co-simulation workflow and the Insite visualisation framework.

Future operators are helped by the major steps taken in the area of CI/CD for the different tools: the co-simulation workflows are trailblazers for a number of EBRAINS wide integration efforts.

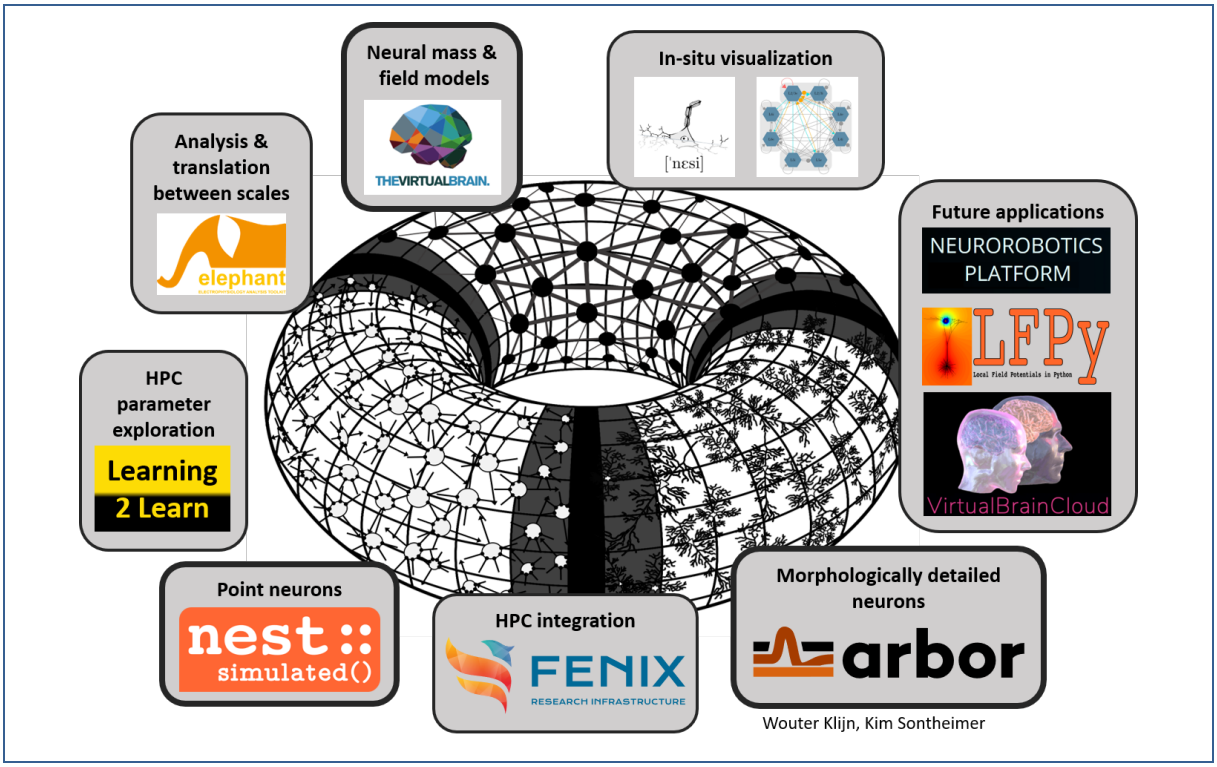


Figure 12: Co-simulation connects and integrates a large set of tools of EBRAINS

3.2.12.1 Recent changes

The work in this phase of the Project has focused primarily on adapting and maturing the coupling mechanisms and workflows developed in the previous phase. This type of work typically does not result in isolated (software) outputs. The work is part of outputs reported in detail elsewhere in this document. The work in this period comprised:

1) Two TVB-NEST co-simulation workflows were developed. One workflow has a strong focus on the science interface and user-interactions. The second workflow is designed for and deployed on HPC systems and further integration with other simulators in mind. The workflows have multiple users.
2) NEST Server was integrated in the standard NEST code-base in a Docker in the Docker hub. NEST Server allows sending and receiving of spikes and recorded data (steering), using a REST API and is being tested in the following workflows:
   a) the NEST Desktop teaching application
   b) HPC back-end for the Neurorobotics Platform[23]
   c) data producer for the in-transit visualisation pipelines (SimDaVis - Insite and SimVisSuite).
3) The new LFPykit software was released. This tool isolates LFP (and related) functionality into a separate module for data exchange with both Arbor and NEURON. The Arbor interface has been extended to allow this integration and is being tested.
4) The Electrophysiology Analysis Toolkit (Elephant) was integrated into three proof of concept workflows:
   a) TVB-NEST co-simulation
   b) as POC the in-transit visualisation pipeline (SimDaVis - Insite) and
   c) as POC in the NEST Desktop teaching tool.

[23] https://neurorobotics.net/

5) The pivot framework for many-to-many spike transport and sorting was integrated into two workflows:
   a) TVB-NEST co-simulation and
   b) the in-situ visualisation pipeline (SimDaVis - Insite).
6) The NEST Desktop client underwent major refactoring regarding the visualisation backend and communication with NEST Server.
7) NESTio, the HPC data transport backend of NEST, is in the final stages of integration into the next NEST version.

3.2.12.2 Outlook

Coming soon:

1) Unification of the two TVB-NEST workflows, HPC deployment of co-simulation using the validated science interface.
2) A 1 million neuron NEST network simulation coupled to TVB and the Neurorobotics Platform.
3) Proof of concept of a NEST Arbor co-simulation model.
4) First phase of CI/CD, multi-scale model provenance tracking and curation, and KG integration.
5) First release of standardised deployment mechanisms (orchestration) of multi-application systems on FENIX and HPC resources.

In the longer-term:

1) Infrastructure for two-way coupling of NEST and TVB on HPC for multiscale simultaneous simulations of whole-brain behaviour, constrained by detailed local behaviour of selected regions.
2) NEST-Arbor multiscale two-way co-simulation infrastructure, enabling the description of higher-level network architecture with NEST and local behaviour of selected regions with Arbor.
3) In-situ infrastructure for data analysis and visualisation, as well as common coupling infrastructure and architecture. Common APIs and standards will be defined and used throughout, to enable the replacement of selected simulators with other simulators (such as NEURON) which implement the required APIs.
4) Infrastructure for LFP forward-model calculations as a one-way co-simulation, which will feed NEST spiking results into a morphologically detailed Arbor simulation, in which LFPykit is used to map transmembrane currents to corresponding extracellular measures of electric activity, such as LFP and EEG.
5) The release of the production version of the orchestration software will be developed to allow scientists to start coupled applications (for one- and two-way co-simulation and analysis) on FENIX, HPC resources and local working machines. For this purpose, a common API has been defined, which the simulators can use, and communication protocols have been specified.

3.2.13 TVB

See Section 3.1.12.

3.3 Data Analysis

3.3.1 Neo

Type: command-line tool or library
Scope: all
Abstraction level: all
Status: mature, available now on HPC, JupyterLab, neuromorphic
Homepage & more information:
- https://neo.readthedocs.io/
- https://kg.ebrains.eu/search/instances/Software/b0d73040-6547-447d-a1be-6761c209973f

Neo is a library for handling neurophysiology data in Python. It provides a standardised representation of electrophysiology and optophysiology data in Python, together with support for reading a wide range of neurophysiology file formats, including Spike2, NeuroExplorer, AlphaOmega, Axon, Blackrock, Plexon andTdt, as well as support for writing to a subset of these formats, plus non-proprietary formats, including NIX and NWB.

Neo improves interoperability between Python tools for analysing, visualising and generating electrophysiology data by providing a common, shared object model. Neo objects behave just like normal NumPy arrays, but with additional metadata, checks for dimensional consistency and automatic unit conversion. Neo is a dependency for several other components of EBRAINS, including Elephant, PyNN, Neo Viewer, the Neural Activity Browser and the in-depth data curation workflow. Neo is a community project with contributions from multiple institutions.

3.3.1.1 Recent changes

Neo v0.9.0 was recently released. New features include a simplification of the data model that is expected to facilitate interoperability with EBRAINS tools and services, as well as improved file handling to broaden the scope of data providers handled by Neo[24].

3.3.1.2 Outlook

The next release of Neo (v0.10.0) will include support for the Neurodata Without Borders (NWB:N) format, an open format that is increasingly used for sharing neurophysiology data, and the SONATA format for simulation outputs. A more efficient representation of spike train data will be added, while maintaining the external API.

The release of Neo v1.0 is planned for 2021. At this point, the API will be considered stable and backwards compatibility maintained. This will be accompanied by a major overhaul of the documentation.

3.3.2 Elephant

Type: command-line tool or library
Scope: all
Abstraction level: all
Status: mature, available now on HPC, JupyterLab, neuromorphic
Homepage & more information:
- https://elephant.readthedocs.io
- https://kg.ebrains.eu/search/instances/Software/f3685e37-cb34-4010-a528-468564482257

[24] https://neo.readthedocs.io/en/stable/releases/0.9.0.html

Elephant is an open-source software Python library that provides analysis methods for activity data from experiments and simulations; in particular, parallel spike trains and time series data of population activity. It is developed as a community open-source project, with contributors from both inside and outside the HBP. Elephant continuously releases new functionality, organised in submodules of the library themed along types of different analyses. These methods constitute a heterogeneous array of tools for the comprehensive investigation of concerted activity from different angles within a single framework. The library is built on top of the Neo library, which provides the underlying data model for the types of activity data analysed by the toolbox. As such, the library's functionality is easily accessible for datasets loaded from a large array of sources, including multiple file formats and simulators.

3.3.2.1 Recent changes

Elephant saw a new major release, 0.8.0, on 7 August 2020 that provides new functionality; in particular, new parallelisation features and the start of a major documentation overhaul, including integrated online tutorials. These tutorials will help scientists and systems engineers in putting example codes using the Elephant library to use quickly. Elephant 0.9.0 was released on 14 November 2020, in preparation for the 2nd Elephant User Workshop. It provides a number of new functions, such as causality estimates or new measures for spike statistics, as well as a number of run-time optimisations. It is also the first Elephant release to pair with the Viziphant prototype. Full details can be seen in the release notes[25].

3.3.2.2 Outlook

In addition, current work in Elephant addresses refactoring and speed optimisations, including GPU testing. These were determined as important for improving the ongoing integration of Elephant analysis functionality into EBRAINS products, such as NEST Desktop. In the same spirit, an initial API is being developed to make Elephant functions accessible within the multi-simulator framework and NEST Desktop, and to facilitate setting up service accounts to perform particular specific analysis pipelines. To facilitate provenance capture, first pilots to record the introspective provenance of results obtained through Elephant were designed and tested. We also expect to see the introduction of more formal data analysis object representations in 2021.

3.3.3 LFPy

Type: command-line tool or library
Scope: single neurons, networks
Abstraction level: biophysical neuron models
Status: mature, available on HPC and JupyterLab
Homepage & more information:
- https://lfpy.readthedocs.io/

LFPy is an open-source Python module for calculating electric and magnetic brain signals from multi-compartment neuron models and networks. It is built on the NEURON simulator, but ongoing work aims to make it function also with other simulators like Arbor.

Capabilities:

- From arbitrary simulated neural activity, LFPy 2.0 can calculate:
  - extracellular action potentials
  - local field potentials (LFP)
  - *in-vitro* MEA recordings
  - electrocorticography (ECoG)

[25] https://elephant.readthedocs.io/en/latest/release_notes.html

- electroencephalography (EEG)
- magnetoencephalography (MEG)

- Convenient Python wrapper around NEURON to control simulations.
- Can calculate EEG signals with simple and complex head models.
- Works for both single cells and large neural networks.

3.3.3.1 Recent changes

LFPy was originally designed as a Python interface to NEURON, with the ability to set up and control NEURON simulations, and to calculate several different measurable brain signals from the simulated neural activity. This means that LFPy's functionality to calculate measurable brain signals was not originally available for other neural simulations, like the NEURON-based BMTK, Brian2 or Arbor.

To mitigate this, and allow for easy integration with other neural simulators, we extracted the parts of LFPy responsible for calculating brain signals into a new (NEURON-independent) Python package called LFPykit[26].

3.3.3.2 Outlook

In collaboration with the Arbor developers, we are currently working on and testing the usage of LFPykit with Arbor, to calculate measurable brain signals from Arbor simulations. We expect to have the first simple examples ready quite soon.

Longer term plans include having many well-documented examples of how to use LFPykit in combination with different neural simulators like BMTK, Brian2 and Arbor, for calculating many different brain signals from single neurons or large neural networks.

3.3.4 TVB

See Section 3.1.12.

3.4 Visualisation

3.4.1 SimVisSuite

Type: desktop app
Scope: single neurons, networks
Abstraction level: point neuron models, biophysical neuron models
Status: prototypes available
Homepage & more information:
Visimpl:
- https://vg-lab.es/visimpl/
- https://kg.ebrains.eu/search/instances/Software/6f8b4c7d-31f6-49b5-a843-2e0b4421be6f

NeuroTessMesh:
- https://github.com/vg-lab/NeuroTessMesh
- https://kg.ebrains.eu/search/instances/Software/4aa7fc48-101c-4f8f-bd6e-559486c133a3

Neuroscheme:
- https://github.com/vg-lab/NeuroScheme
- https://kg.ebrains.eu/search/instances/Software/a5f69883-eb34-4dd5-bc44-30e5d4718f27

[26] https://github.com/LFPy/LFPykit

The Simulation Data Visualisation Suite is a collection of visualisation tools offering interactive visual analysis capabilities for network and neuron-level simulations. It is comprised of the tools ViSimpl, NeuroTessMesh and NeuroScheme.

The framework we have developed provides different views, allowing the exploration of neuronal morphology, neuronal activity, and neuronal connectivity. Since the brain is inherently a multiscale system, we provide interactive visualisations at different levels of abstraction. This multilevel approach helps in managing complexity and linking phenomena taking place at different organisational levels.

The framework developed additionally allows using complementary representations of the same data (realistic and symbolic). Realistic representations provide detailed visualisations of the dataset under analysis, while symbolic representations provide simplified overviews, where different areas can be represented at lower levels of detail in order to facilitate focusing user attention on specific features or areas of interest.

3.4.1.1 ViSimpl - Study simulation data and their associated spatial and temporal features by exploratory visual analysis

ViSimpl involves two components: SimPart and StackViz. SimPart is a three-dimensional visualiser for spatio-temporal data that allow spatio/temporal analysis of the simulation data, using particle-based rendering. StackViz illustrates how the electrophysiological variables evolve over time and provides a temporal representation of the data at different aggregation levels. Together, they allow users to visually discriminate the activity of different groups of neurons, and provide detailed information about individual neurons of interest. These components share synchronised playback control of the simulation being analysed and work together as linked views, although they are loosely coupled and can also be used independently. Figure 13 shows an example for the visualisation of activity using ViSimpl.

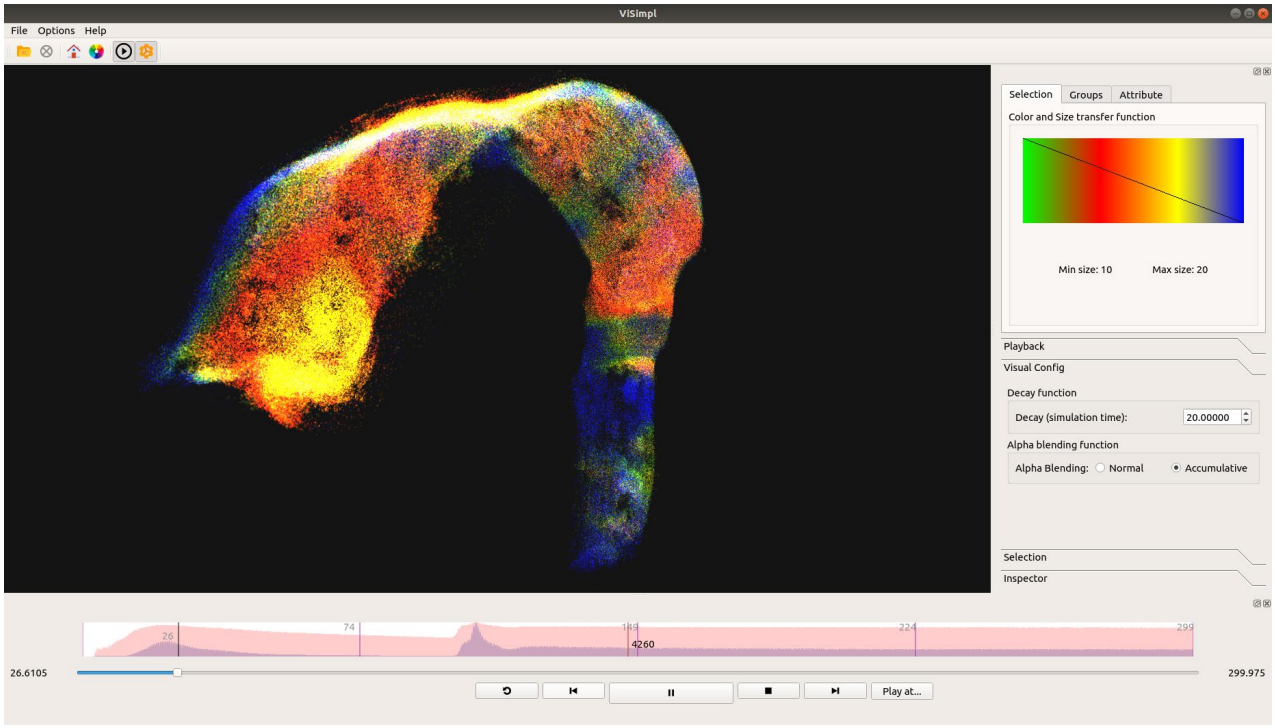


Figure 13: Visualisation of activity in a hippocampus model using ViSimpl

The model is simulated by Michele Migliore on behalf of the HBP Hippocampus Team (EPFL, CNR, IEM HAS, UCL) using the GALILEO HPC system at CINECA.

3.4.1.2 NeuroTessMesh - Visual Analysis of Neuron Anatomy

NeuroTessMesh allows users to visualise neurons and neural circuits consisting of a large number of cells with on their desktops. It enables the visualisation of the 3D morphology of cells included in

open databases, such as NeuroMorpho, and provides the tools needed to approximate missing information such as the soma's morphology. NeuroTessMesh takes morphological tracings of cells acquired by neuroscientists as its only input. It generates 3D models that approximate the neuronal membrane. The resolution of the models can be adapted at the time of visualisation. NeuroTessMesh can assign different colours to different parts of a morphology, in order to visually codify relevant morphological variables, or even neuronal activity. Figure 14 shows an example for the visualisation of the detailed morphology of neurons using NeuroTessMesh.

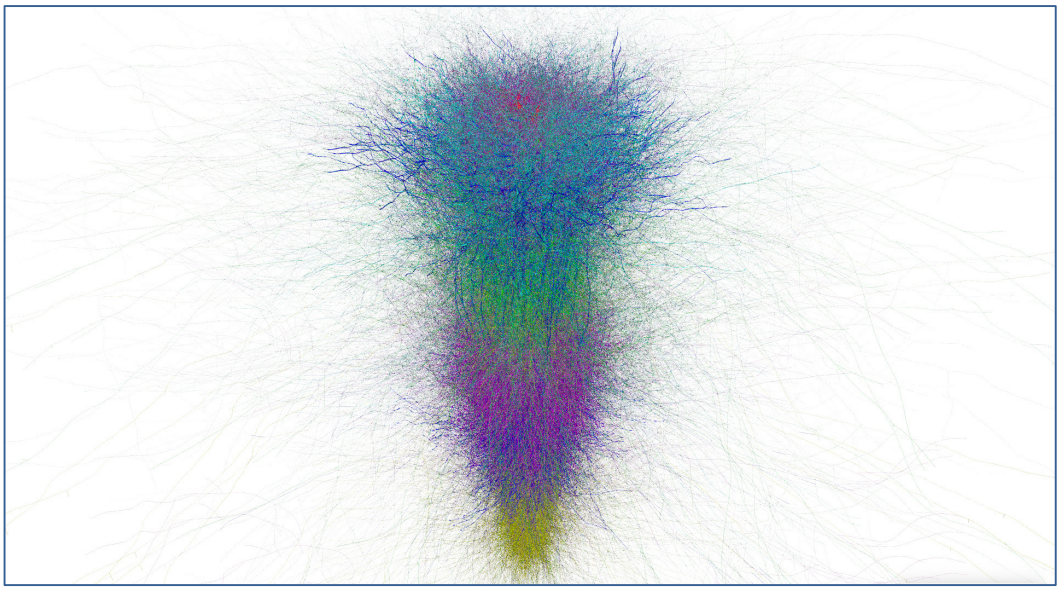


Figure 14: Visualisation of the detailed morphology of neurons in a cortical column

Each layer is shown in a different colour.

3.4.1.3 NeuroScheme - Abstract representation of Neural Networks

NeuroScheme uses schematic representations, such as icons and glyphs, to encode attributes of neural structures (i.e. neurons, columns, layers, populations, etc.). This abstraction alleviates problems with displaying, navigating and analysing large datasets. NeuroScheme has been designed specifically to manage hierarchically organised neural structures; one can navigate through the levels of the hierarchy, and home in on their desired level of details. Figure 15 shows the NeuroScheme interface.

NeuroScheme works using what we call "domains". These domains specify which entities, attributes and relationships are going to be used for a specific use case. NeuroScheme currently has two built-in domains: "cortex" and "congen". The "cortex" domain is designed for navigating and analysing cerebral cortex structures (i.e. neurons, micro-columns, columns, layers, etc.). The "congen" domain can be used to define the properties of both cells and connections, create circuits composed of neurons, and build populations. Groups of populations can be easily moved to a higher level of abstraction (such as column or layer), allowing one to create complex networks with little effort. These circuits can be exported afterwards and used for further analysis and simulations.

More information can be found in the publication by Velasco *et al*. 2020[27].

[27] Velasco I, Toharia P, Benavides-Piccione R, Fernaud-Espinosa I, Brito JP, Mata S, DeFelipe J, Pastor L, Bayona S (2020) Neuronize v2: Bridging the Gap Between Existing Proprietary Tools to Optimize Neuroscientific Workflows. Front. Neuroanat. 14. (P2620)

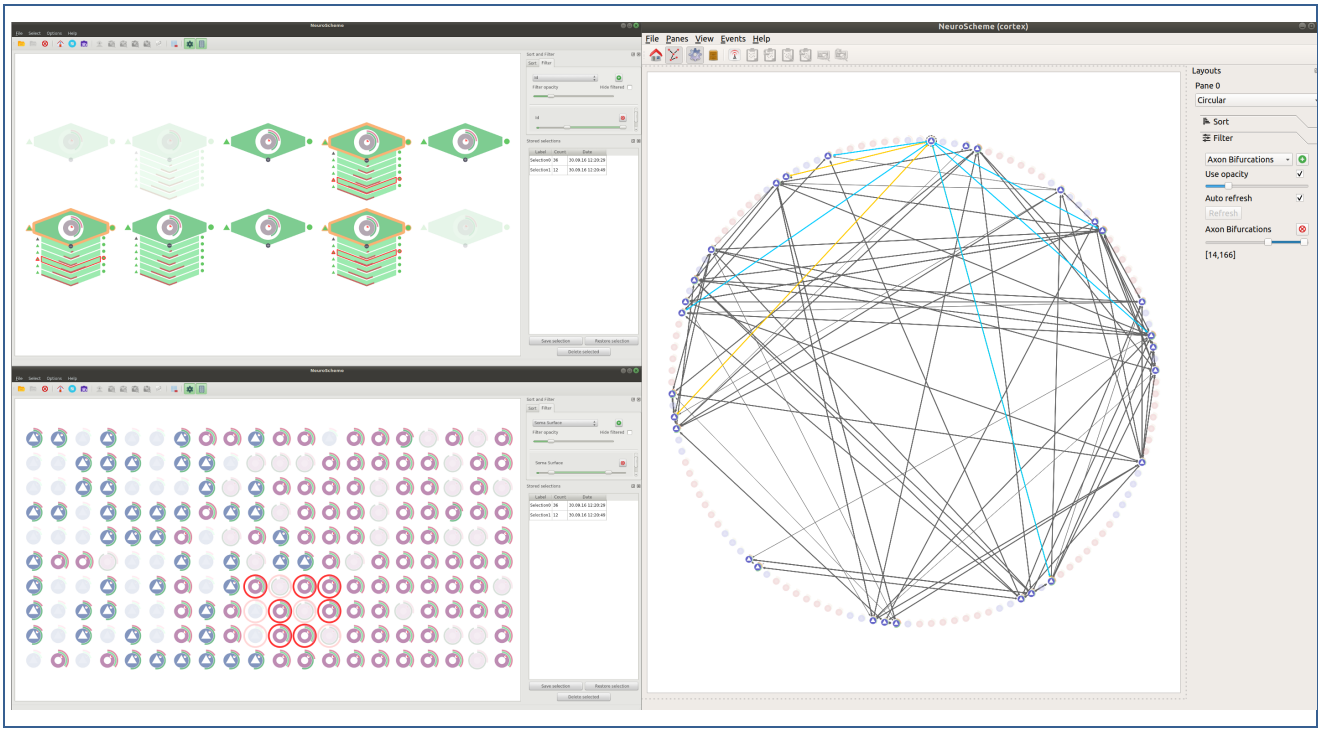


Figure 15: Visualisation of cortex structures and of a circuit in the "congen" domain

On the left: visualisation of structures of the cortex at minicolumn abstraction layer (top left) and at neuron abstraction level (bottom left). On the right: visualisation of a circuit in the "congen" domain.

3.4.1.4 Recent changes

Due to delays caused by COVID issues, only minor changes have been introduced during this period.

3.4.1.5 Outlook

The testing and integration of these tools into the EBRAINS infrastructure is planned for 2021. ViSimpl is the first candidate to be integrated into to the new *in-situ* pipeline design in close cooperation with their developers. It is planned that all tools will be available in easy-to-deploy containerised packages (this work is ongoing in the case of ViSimpl, NeuroTessMesh and NeuroScheme).

3.4.2 NEST Desktop

Type: web app
Scope: networks
Abstraction level: point neuron models
Status: mature, available as standalone app
Homepage & more information:

- https://nest-desktop.apps.hbp.eu/#/
- https://nest-desktop.readthedocs.io
- https://kg.ebrains.eu/search/instances/Software/b1f7faf5-aafe-4675-a1cb-532326f10b5c

The main goal of NEST Desktop is to offer an install-free, interactive visual tool for classroom use in computational neuroscience courses. NEST Desktop is a web-based GUI application for NEST Simulator. NEST Desktop enables the rapid construction, parametrisation and instrumentation of neuronal network models. It offers interactive tools for construction of visual networks, running simulations in NEST and applying visualisation to support the analysis of simulation results.

NEST Desktop consists mainly of two views and a connection to a server-based NEST instance, which can be controlled using the web-based NEST Desktop front-end (Figure 16). The first NEST Desktop view enables the user to create point neuron network models interactively. A visual modelling

language is provided, and a simulation script is automatically created from this visual model. The second view enables the user to analyse the returned simulation results using various visualisation methods. NEST Desktop also offers additional functionality, such as employing Elephant for more sophisticated statistical analyses.

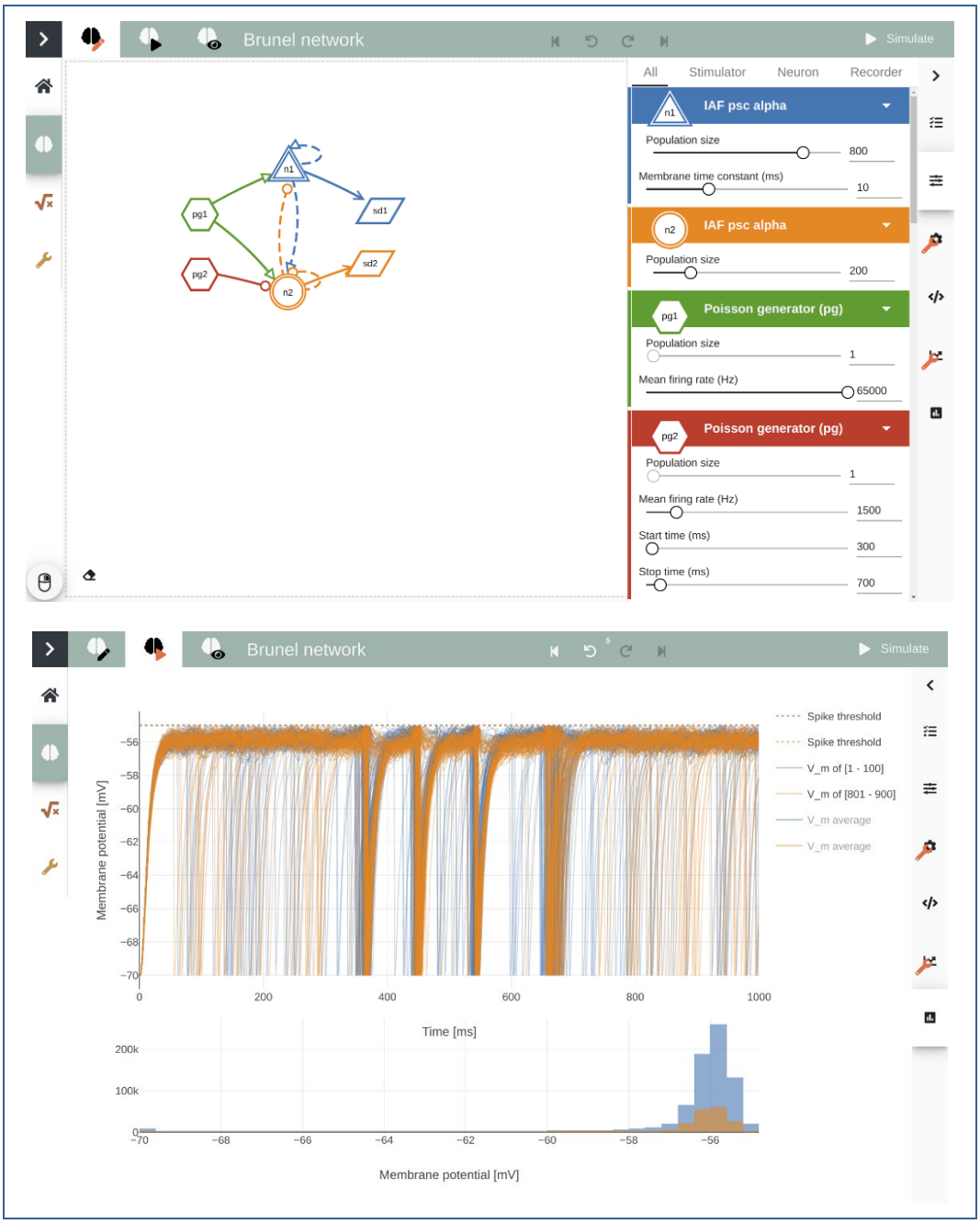


Figure 16: Screenshots of NEST Desktop in modelling (top) and visual analysis (bottom) mode

Interactively and visually modelled neural networks can be simulated by one click and easily analysed via various types of interactive visualisations.

3.4.2.1 Recent changes

The current major release (2.5) was published on GitHub in October 2020. NEST Desktop is also usable through EBRAINS via this link: https://nest-desktop.apps.hbp.eu/#/.

We used NEST Desktop in the summer of 2020 for an online course on computational neuroscience, with around 25 students at Freiburg University. We received positive feedback on the tool and its applicability. In an online survey, we received positive qualitative feedback, as well as good results for usability (SUS score of 58) and user experience (UEQ around 1.0 on all sub-scales) in quantitative evaluations.

3.4.2.2 Outlook

In the next steps, NEST Desktop will be ported to the open source framework Vue.js to increase its performance. Furthermore, NEST Desktop will be integrated with InSite to enable connection to NEST via this middleware and thus run and analyse larger networks.

3.4.3 SimDaVis - Insite

Type: command-line tool or library
Scope: networks
Abstraction level: point neuron models, biophysical models
Status: prototype available
Homepage & more information:

- https://vrgrouprwth.github.io/insite
- https://kg.ebrains.eu/search/instances/Software/eb0b86a5-eb49-4a87-9128-6ca592293218

Insite is a library that integrates with existing neuronal network simulators and provides a REST API to access the data generated by the simulation while it is running. This includes the generated spikes of the neurons, as well as other measured attributes (e.g. membrane potentials). In addition, it provides access to the network structure, such as the number of neurons and their models and parameters.

The framework is meant to be used by third-party visualisation and analysis tools to provide a direct view into a running simulation and, for instance, to shorten the time to detect errors. It is the first step towards interactive steering of simulators.

3.4.3.1 Recent changes

We greatly simplified the way users interact with the pipeline, by changing the API to a REST based architecture using HTTP requests and a standardised JSON data format. We have a release candidate for the new pipeline which was developed using feedback from potential users. User documentation on how to setup the pipeline, as well as the API, is now publicly available and easy to access. Additionally, we provide Docker containers on Docker Hub for easy deployment and testing of the software.

3.4.3.2 Outlook

In the short term, we plan an official first release of the pipeline after the restructuring process. This will include full support for the NEST simulator. We then want to add support for the Arbor simulator to the pipeline with the next update. We are currently also working on the integration of analysis functionality directly into the pipeline, enabling in-transit-analysis of neural network simulations. Within the next 12 months, we want to have a prototype implementation with a defined API ready.

The major aspect of the long-term plans is full integration into the EBRAINS infrastructure, to further simplify the usage of the pipeline. This integration would allow users to start simulations remotely on EBRAINS-managed HPC resources and connect to them using the specified REST API for analysis and visualisation purposes. This will most likely include the adaptation of the NEST-server functionality recently added to the NEST simulator. Additionally, access to data from different abstraction levels is provided, by integrating TVB into the Insite pipeline. Furthermore, we plan to integrate basic steering capabilities into the pipeline to enable interactive supercomputing. This depends greatly on the steering functionality provided by the simulator and we cannot estimate this yet.

3.4.4 Elephant Visualization (Viziphant)

Type: command-line tool or library
Scope: all
Abstraction level: all
Status: prototype available
Homepage & more information:
- https://viziphant.readthedocs.io
- https://github.com/INM-6/viziphant

Elephant Visualization (Viziphant) represents a complementary library for Elephant to provide functionality related to visualisation of data and analysis results. These graphing functions are deliberately decoupled from the Elephant library to keep the code of the Elephant easier to maintain. They aim to assist scientists in quickly visualising analysis results, as well as helping to keep Elephant tutorials easy to read by removing clutter from the graphics creation. In the long term, we envision that Viziphant will provide simple illustrations of all Elephant analysis functions, based on standard Python plotting libraries.

3.4.4.1 Recent changes

Viziphant now includes a number of plotting routines that were used in the online tutorial series created for the 2nd Elephant User Workshop.

3.4.4.2 Outlook

Along with our efforts to improve the Elephant documentation through accompanying tutorials, we will continuously add new visualisations for the individual Elephant functions. For 2021, we plan to streamline the design of the visualisation and provide a detailed portfolio of tools to plot the underlying activity data provided in the Neo format. Discussions on feasible integrations of Viziphant in the EBRAINS service portfolio are ongoing.

3.5 Validating models against experimental data

3.5.1 Validation test libraries

Type: library
Scope: all
Abstraction level: all
Status: mixed. Some mature, available on HPC, JupyterLab, others in progress
Homepage & more information:
NetworkUnit:
- https://github.com/INM-6/NetworkUnit
- https://kg.ebrains.eu/search/instances/Software/e3566d6b-fed9-4f5d-b704-04e6eda76c99

MorphoUnit:
- https://github.com/appukuttan-shailesh/morphounit
- https://kg.ebrains.eu/search/instances/Software/860c230e-56cf-4f49-bdb7-cd438a3de0e2

HippoUnit:
- https://github.com/KaliLab/hippounit
- https://kg.ebrains.eu/search/instances/Software/fd369ab8-c788-4c7b-89ec-594aa3e46c24

CerebUnit:
- https://cerebtests.readthedocs.io/

Validation Framework - Python Client:
- https://hbp-validation-client.readthedocs.io/en/master/#
- https://kg.ebrains.eu/search/instances/Software/c053931f-ddd1-4416-8b3e-5506677c4bda

These comprise a collection of Python libraries to support structured model validation. All of the libraries are based on the SciUnit framework, developed by the Crook-Gerkin Lab at Arizona State University (members of the HBP Partnering Project "SciUnit"). This framework allows model implementations and validation test implementations to be decoupled by declaring standardised interfaces. This decoupling means that different models of the same structure or phenomenon can be easily validated against the same experimental datasets, and allows us to build up libraries of tests. Our goal is that the majority of models available through KG search will have direct links to the results of automated validation tests, making it easy for EBRAINS users to understand to what extent a given model has been validated. The current tests are grouped into separate Python libraries, according to their domain of application: either the brain region being modelled, or the scope of the model (e.g. single neuron, local network, brain region, etc.)

3.5.1.1 Recent changes

Incremental updates have been made to several of the existing test suites, such as CerebUnit. Work on some validation suites like SynapseUnit, conceptualised during end of SGA2, have made initial releases. A manuscript dealing with HippoUnit, another validation suite, has been submitted for publication.

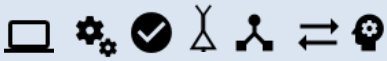
3.5.1.2 Outlook

We plan to extend the various validation suites, and publish results derived from these. Meanwhile, in addition to creating more validation tests at the level of single cells – different tests based on cell type – work has been initiated on creating tests for synapses between cells. This will then be followed by creating tests for subcellular mechanisms. CerebUnit will also be extended to measure sensitivity, specificity and accuracy of the validation test. We also intend to look into developing newer test suites, such as for testing whole brain models. We are currently also working on a study to demonstrate how models can be compared using the VF tools. The NetworkUnit library is currently being equipped with additional tests and metrics and also received improved compatibility to the NEST simulation framework.

We intend to disseminate the utility of incorporating the validation services in the modelling and reviewing workflows, via publications, hackathons and workshops on model validation that help familiarise the community with the test suites and their usage, and promote community-driven development of model validation test suites.

3.5.2 Model Catalog

Type: web app, REST API
Scope: all



Abstraction level: all
Status: mature, available now as standalone web app REST API, EBRAINS integration almost complete
Homepage & more information:

- https://model-catalog.brainsimulation.eu/docs/
- https://validation-v2.brainsimulation.eu/docs

The HBP Model Validation Service provides web-based tools for working with computational models and validating such models against experimental data. It consists of a REST web service and two clients: the Model Catalog app and a Python client. Underlying metadata are stored in the KG. The service allows users to create, edit, search and view metadata about models and validation tests, and to register, search, view and compare the results of validation tests. Figure 17 shows a screenshot of the Model Catalog app.

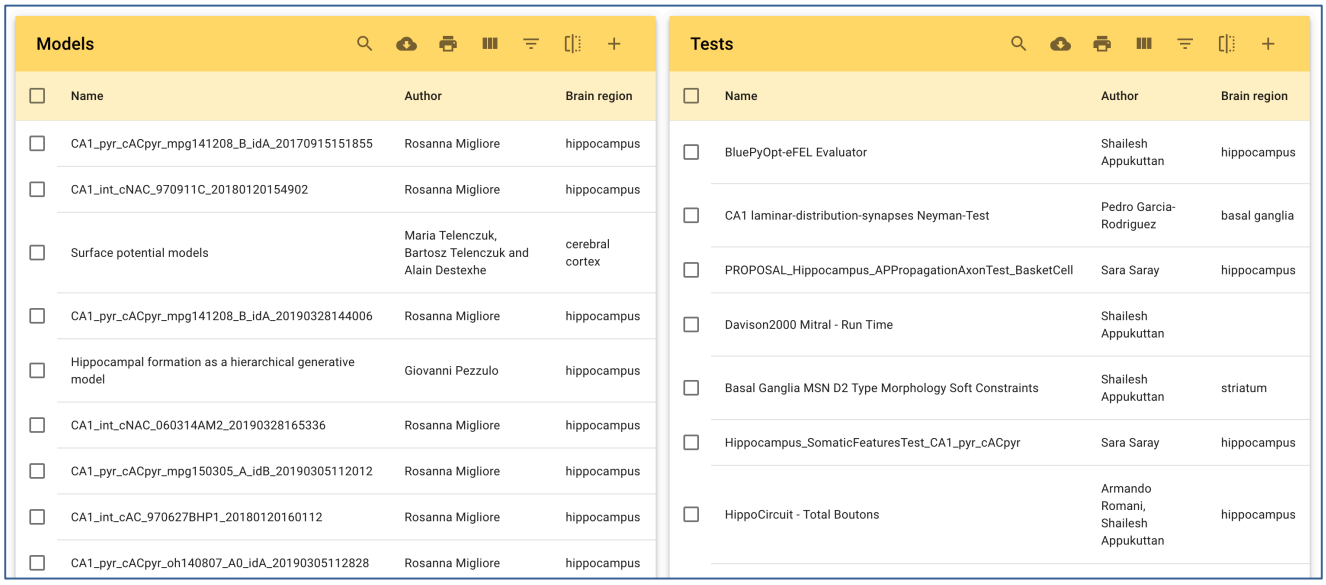


Figure 17: Screenshot of the Model Catalog app

3.5.2.1 Recent changes

Both the REST web service and the Model Catalog app have been completely rewritten to facilitate maintainability, to use more modern technologies (e.g. ReactJS in place of AngularJS), and to simplify the migration to EBRAINS authentication. Both the web service and the app now make use of EBRAINS authentication, the EBRAINS Drive and the EBRAINS KG.

Version 1 of the Model Catalog app is still available in the Collaboratory v1, but its use is deprecated in favour of version 2.

3.5.2.2 Outlook

The Model Catalog is currently available as a stand-alone app; it will soon be available in addition as a Collaboratory (v2) app, which will facilitate collaboration on model validation.

With the upcoming release of version 3 of the EBRAINS KG, the web service will need to be updated accordingly. Performance improvements for the web service are planned. Both the web service and app will be extended to link models with simulations generally, rather than only those simulations performed for model validation, as at present.

3.5.3 Model building workflows

Type: workflows / Jupyter Notebooks
Scope: all
Abstraction level: all
Status: mature, available in Collaboratory v1, EBRAINS integration in progress
Homepage & more information: https://collab.humanbrainproject.eu/#/collab/1655

A large number of exemplar workflows have been developed, based on Jupyter notebooks, for building data-driven single neuron and network models, and are available in Collaboratory v1. They include:

1) Rebuild an existing single hippocampal cell model[28]
2) Build your own single hippocampal cell model using HBP data[29]
3) Synaptic events fitting[30]

3.5.3.1 Outlook

The three workflows listed above will be migrated to EBRAINS (Collaboratory v2) in the coming months.

4. Behind the scenes

The tools, services and workflows described above depend upon, are integrated with, and/or make use of many other EBRAINS components, middleware, and infrastructure services. In the cases noted above, this integration is incomplete and will be completed within the next two years. The principal dependencies are:

- Identity and Access Management (IAM) service
  - EBRAINS services that provide access to private or sensitive data, or to limited computing resources require users to have an EBRAINS account and to log-in via a single sign-on scheme.
- JupyterLab Service
  - Many workflows are run in Jupyter notebooks, running on the *Virtual Machine Services* (see below), and with access to *Interactive Computing Services* via the UNICORE API and to *Neuromorphic Computing Services* via the Neuromorphic Job Queue API.
- Collaboratory / Drive storage
  - Collaboration between multiple users is enabled by shared workspaces, or "Collabs". The Collaboratory also provides shared storage, called the Drive, which can be accessed in Jupyter notebooks, through a web browser interface, and/or synchronised with a user's own computer.
- Knowledge Graph
  - The KG is the central metadata store for all of EBRAINS, and provides both graphical user interfaces and API access.
- Provenance Service
  - Still at the prototype stage, the Provenance Service API allows users to track their activity using EBRAINS services, and automatically captures the metadata needed to ensure reproducibility.
- Curated data in Archival Data Repositories
  - Many modelling and simulation tools, services and workflows make use of datasets in the EBRAINS Data Repository, which have their metadata in the KG and their data files in the EBRAINS Archival Data Repositories.
- Neuromorphic Computing Services

[28] https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/single_cell_building/hippocampus/rebuild_existing/rebuild_scm.html
[29] https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/single_cell_building/hippocampus/build_scm_hbp/build_scm_hbp.html
[30] https://humanbrainproject.github.io/hbp-sp6-guidebook/online_usecases/trace_analysis/syn_events_fit/syn_events_fit.html

- Simulations of spiking neuronal network models can be performed on the bio-inspired SpiNNaker and BrainScaleS systems, which offer several advantages over traditional high-performance computing, such as energy efficiency and accelerated operation.

- Virtual Machine Services
  - The five European supercomputer centres within EBRAINS provide virtual machines using the OpenStack platform. These VMs are used to run most of the web apps, web services and middleware provided by EBRAINS.
- Scalable Computing Services
  - The supercomputer centres also provide access to HPC resources, for applications requiring a high level of parallelisation, high-memory usage and/or multiple-GPU access. EBRAINS users can request individual access to these resources, but several modelling and simulation services make use of these resources on behalf of users without an explicit access request being needed.
- Interactive Computing Services
  - Quick access to single compute servers to analyse and visualise data interactively, or to connect to running simulations using the scalable compute services.

The interdependencies between the different tools and services are summarised in Table 2.

Table 2: Interdependencies between EBRAINS tools and services

		Other SCs										SC3																
		IAM	JupyterLab	Collaboratory/Drive	KG	Prov Service	Curated Data	Neuromorphic	VMs	Scalable HPC	Interactive HPC	Model Catalog	Electrical model building toolset	PyNN	NESTML	Snudda	Brain Scaffold Builder	NEST	Arbor	NEURON/coreNEURON	Multiscale, multi-simulator, co-simulation	Neo	Elephant	LFPy	Elephant Visualization (Viziphant)	Validation test libraries	SimDaVis - Insite
app	Subcellular webapp																											
app	Feature Extraction Tool																											
app	Hodgkin-Huxley Neuron Builder																											
app	NEST Desktop																											
app	The Virtual Brain (TVB)																											
app	SDA - Simulation of Diffusional Association (and webSDA webserver)																											
app	ArDocK - Webserver for prediction of protein interaction surfaces																											
app	CGMD Platform - Web server for running coarse-grained molecular dynamics simulations																											
app	SSB Platform / METNET - Web server for kinetic modeling for neurotransmitters																											
app	MoDEL-CNS - Web server for molecular dynamics simulation data on signal transduction																											
app	Small Circuit In Silico Experiments Tool (Rat Hippocampus CA1)																											
app	Brain Areas Circuit In Silico Experiments Tool (Rat Hippocampus CA1)																											
app	Neuron as a Service (aka BlueNaaS aka Single Cell in silico Experiment tool)																											
app	SimVisSuite																											
app	SimDaVis - Insite																											
app	Neuromorphic Computing Job Manager app																											
app & service	Model Catalog																											
library	Electrical model building toolset																											
library	PyNN																											
library	NESTML																											
library	Snudda																											
library	Brain Scaffold Builder																											
library	NEST																											
library	Arbor																											
library	NEURON/coreNEURON																											
library	Multiscale, multi-simulator, co-simulation																											
library	Neo																											
library	Elephant																											
library	LFPy																											
library	Elephant Visualization (Viziphant)																											
library	Validation test libraries																											
app (Jupyter)	Rebuild an existing single hippocampal cell model																											
app (Jupyter)	Build your own single hippocampal cell model using HBP data																											
app (Jupyter)	Synaptic events fitting																											
library/workflow	Subcellular model building toolset																											
workflow/app	Jupyter notebook for computing dissociation rates using tauRAMD method																											
workflow/app	JN and scripts for using molecular level tools (to port from HBP to Ebrains platform)																											
library	molecular level tools (apps and scripts), e.g. MD-IFP, PIPSA																											

Green indicates that the tool or service in the left-hand column uses, or can easily be run on, the tool or service listed above. Yellow means that such integration is planned, partial, or in progress.

5. Discussion

This report presents the current and planned EBRAINS tools, services and workflows for brain modelling and simulation, and their implementation status as of late 2020. Many of these components were already integrated into the HBP platforms, and into the first version of the Collaboratory. The migration of these components to EBRAINS is intended to improve maintainability and sustainability, and to yield a more coherent, better integrated user experience.

The tools cover different aspects of the simulation life cycle (Figure 1). Some of the tools are stand-alone products of generic use, as for example the Elephant data analysis library, which handles both experimental and simulation data, or simulation engines such as Arbor, NEST, and TVB, which within their respective modelling scopes allow for great flexibility with respect to model definition. Neo and SONATA facilitate interoperability in terms of data formats. Some of the simulators already support the SONATA network-description format, others are in the process of enabling support. Graphical user interfaces (GUI) to the simulation engines (e.g., Subcellular web app, NEST Desktop, TVB GUI) integrate the simulators into workflows adding functionality (e.g., model creation, data analysis) and usability. Some of the tools combine into tailored workflows for simulations of specific realistic models of neuronal systems integrating experimental data (e.g., Hodgkin-Huxley Neuron Builder, Feature Extraction Tool, Single Cell *In Silico* Experiment Tool). Some of the tools provide horizontal links, such as PyNN, which offers a single coherent interface to different simulators (e.g., Brian, NEST, NEURON) and neuromorphic systems (SpiNNaker, BrainScaleS) allowing for cross-validation.

In general terms, the steps needed to integrate HBP Platform applications and workflows into EBRAINS are:

- migration from v1 to v2 of the IAM service;
- migration of metadata to the KG;
- deployment on the Virtual Machine services provided by the HPC Partners;
- use of Drive storage for intermediate data in place of Collaboratory v1 storage;
- retrieval of curated data, where needed, from the Archival Data Repository;
- improved documentation, migration of Collab-based documentation to Collaboratory v2;
- for workflows in Jupyter notebooks: migration from Python 2.7 to Python 3.6+.

As described above in Section 3, some apps and workflows have completed all of these stages, others are part way there, and others are still in the planning stage. By the end of 2022, all the tools, services and workflows described in this report are expected to have completed their integration into EBRAINS.

Further steps, leading to deeper integration and an improved user experience, will follow as new infrastructure and middleware services become available, for example:

- recording of provenance information through the Provenance API;
- standardised, automated deployment with in-built quality control checks through a Continuous Integration/Continuous Deployment process;
- improved monitoring and responsive scaling of applications and services;
- more powerful tools for multiscale, multi-application workflows that require orchestration of multiple tools and services.

5.1 Comparison to/interactions with related tools and services

We might ask how the EBRAINS modelling and simulation tools, services and workflows compare to the state-of-the-art, i.e. to other digital research platforms for computational neuroscience and bio-inspired machine learning.

In computational neuroscience, and in systems biology applied to neuroscience, a number of web-based databases and simulation platforms exist.

ModelDB[31] is developed at Yale University and is currently funded by the US National Institute for Deafness and other Communication Disorders. It is well established as a resource for model sharing in computational neuroscience. We are informally collaborating with the ModelDB developers to improve interoperability. For example, we plan to cross-link models between ModelDB and the EBRAINS Model Catalog, so that, for example, ModelDB users will be able to find and run simulations of ModelDB models on EBRAINS.

OpenSourceBrain[32] (OSB) is developed at University College London and funded by the Wellcome Trust. It provides a model repository built on GitHub and several graphical tools for simulating single neurons and network models in the browser, using the NEURON simulator. Larger simulations on the platform can make use of the Neuroscience Gateway (see below). These tools are comparable to the following EBRAINS applications: Hodgkin-Huxley Neuron Builder; Small Circuit *In Silico* Experiments Tool, Brain Areas Circuit *In Silico* Experiments Tool and Neuron as a Service. We are informally collaborating with the OSB developers to improve interoperability. For example, we plan to cross-link models between the EBRAINS Model Catalog and the OSB repository.

BioModels[33] is a repository for mathematical models of biological systems, not just neuroscience ones, and focused primarily on subcellular/molecular modelling. BioModels is supported by the European Bioinformatics Institute and is part of the ELIXIR infrastructure. BioModels is complementary to the Subcellular web app and model building toolset (Sections 3.1.3 and 3.1.4) and subcellular models in BioModels can be used as input for these tools.

The Blue Brain Portal[34] provides a number of databases (e.g. Channelpedia, Blue Brain Cell Atlas and the Neocortical Microcircuit Collaboration Portal), together with links to the HBP/EBRAINS tools to which the Blue Brain Project contributes. Since the Blue Brain Project is a Partner in the Human Brain Project, the level of interoperability between BBP-specific and EBRAINS tools is high.

The Neuroscience Gateway[35] aims to facilitate access by computational neuroscience researchers to US National Science Foundation-funded HPC resources. It provides free computer time to neuroscientists, with access via a web portal or programmatically via a REST API. The simulation engines provided are largely the same as those on EBRAINS: NEURON, NEST, PyNN, etc.

The Allen Institute for Brain Science provides several databases through the Allen Brain Map[36] portal.

At present, EBRAINS services and those described above are more complementary than in competition. For example, the Neuroscience Gateway provides simplified access to US-funded HPC resources while EBRAINS provides the same for European HPC resources through the ICEI project. Similarly, the field will be best served by promoting interoperability between ModelDB, OpenSourceBrain and EBRAINS.

In terms of simulation engines and related tools, EBRAINS mostly works in collaboration with the wider community. NEST and PyNN, for example, are both developed as community projects, with contributions from both within and outside the HBP. NEURON is a well-established project with its own NIH funding and, increasingly, community contributions, but it is also well integrated into

[31] http://modeldb.science/
[32] https://www.opensourcebrain.org/
[33] https://www.ebi.ac.uk/biomodels/
[34] https://portal.bluebrain.epfl.ch/
[35] https://www.nsgportal.org/
[36] https://portal.brain-map.org/

EBRAINS. CoreNeuron, a performance-focused reimplementation of the central simulation code of NEURON, is a collaboration between the HBP, the Blue Brain Project and the NEURON core developers. Other simulators developed outside of EBRAINS, such as Brian, GeNN, ANNarchy, or MOOSE, can nevertheless be used within the EBRAINS infrastructure (and are used by many of the scientists within the HBP), and there is ongoing work to improve their integration with EBRAINS; for example, PyNN interfaces for Brian v2 and GeNN are under development.

5.2 Conclusions

The goal of EBRAINS modelling and simulation services is to enable computational neuroscience researchers to perform cutting edge research at multiple scales, from molecular/subcellular to whole brain, across scales and at different levels of abstraction. Integrating the different simulation engines, analysis and visualisation tools, and graphical interfaces needed to perform such research into a cohesive infrastructure enables the conception and execution of sophisticated and reproducible workflows.

This integration work is the primary focus of this final phase of the HBP, although improvements to individual components still continue. As can be seen in the reports in Section 3, good progress is being made:

- Web services and apps are migrating from the original HBP authentication/authorisation system to the EBRAINS system, with benefits that include improved maintainability.
- Services are being moved from local and cloud hosting to the EBRAINS virtual machine service,
- The migration from version 1 to version 2 of the Collaboratory,
- Adoption of the Knowledge Graph as the primary source of metadata, and
- Use of EBRAINS Archival Data storage as the primary source of curated data.

As this work proceeds over the next two years, more and more services will become available through the EBRAINS portal (https://ebrains.eu). Project-wide procedures for quality control, continuous integration and continuous deployment will help to raise the maturity and stability of all components and the overall infrastructure. Project-wide tools and mechanisms for end users to design and execute automated computational workflows, using HPC or neuromorphic computing resources as needed, and to capture the provenance of workflow executions to support reproducibility and scientific audit of results, will be brought to maturity.

The significance of the progress made over the past nine months is that we have begun to demonstrate that the cutting-edge apps, libraries and workflows developed by the HBP are not a disparate collection of technologies, but rather together form a cohesive and flexible infrastructure for 21st century digital science.