20 years of NEST: a mature brain simulator

Jochen Martin Eppler 1, Susanne Kunkel 2, Moritz Helias 1, Yury V. Zaytsev 2, Hans Ekkehard Plesser 3, Marc-Oliver Gewaltig 4, Abigail Morrison 1,2, Markus Diesmann 1

1 Institute of Neuroscience and Medicine (INM-6), Institute for Advanced Simulation (IAS-6), Jülich Research Centre and JARA, Jülich, Germany
2 Simulation Lab Neuroscience, Jülich Supercomputing Centre, Jülich Research Centre, Jülich, Germany
3 Department of Mathematical Sciences and Technology, Norwegian University of Life Sciences, Ås, Norway
4 Blue Brain Project, École polytechnique fédérale de Lausanne EPFL, Lausanne, Switzerland

Contact: j.eppler@fz-juelich.de

Introduction

Efficient and reliable simulation tools are essential for progress in brain research. A wide range of simulators has been developed, each specialized on one or few spatial and temporal scales [1]. But the reproducible simulation of complex systems such as the brain is a very demanding challenge.

Thus, the Computational Neuroscience community concentrated on a few reliable and widely used simulation tools in recent years. This concentration was not least the result of a series of large-scale EU funded projects, such as FACETS, BrainScaleS and the recently announced Human Brain Project.

The Neural Simulation Tool NEST [2] saw its first incarnation in 1993. By tightly coupling software development with research in Computational Neuroscience, simulator technology evolved steadily and NEST is a powerful simulation tool for brain-scale simulations today.

Tools for modern Neuroscience research


Precise spike timing enables one to study synchronization in networks without side-effects caused by using a discrete time grid, on which the spikes are forced during the simulation [10].

Correctness and release stability

A battery of unit tests helps to make sure that all parts of NEST are working as expected both in serial and parallel scenarios. For neuron and synapse models, the tests compare simulated results to the analytical solutions [14].

Continuous integration (CI) techniques allow to build NEST with different compiler and library options upon every check-in and thus increase release stability and correctness [15].

Regular open source releases under the terms of the GPL give the users the chance to use and test the newest technology, while developers get feedback via bug reports and mails on the NEST mailing list. The releases are available on the homepage of the NEST Initiative at http://www.nest-initiative.org.

PyNEST is the standard interface to NEST. It allows to specify
- complex stimuli and stimulation patterns
- the organization and connectivity of the network and
- data analysis in a single programming language [3].

PyNN is a simulator-independent description language and an interface for Python that supports many simulators and controls them through simulator-specific backends [4].

Figure 1: The different software layers, which make up the Python UI subsystem of NEST. Users can access all of the layers from their simulation scripts.

The topology module and an interface to the CSA support the user with the construction of structured networks and spatial connectivity patterns [11, 12].

A MUSIC interface allows to couple NEST to other simulators at runtime and send and receive spikes as well as continuous variables to bridge multiple scales, and to bind NEST to ones own program code [13].

References


Acknowledgements: The development of NEST has been made possible by the support of many institutions, including Weizmann Institute, Ruhr-Universitat Bochum, Universität Freiburg, Honda Research Institute Europe, Max-Planck-Institute for Fluid Dynamics, Norwegian University of Life Sciences, RIKEN Brain Science Institute, Bernstein Center for Computational Neuroscience Freiburg, Helmholtz Gesellschaft and FZ Jülich, EPFL and BlueBrainProject, EU grants FACETS and (FP6-15879) BrainScaleS (FP7-269921), Research Council of Norway grant eNeuro (178892/V30).