Human Brain Project

EBRAINS

Tutorial:

Modeling Networks with NEST

Hans Ekkehard Plesser

Dept of Data Science, Norwegian University of Life Sciences

Institute of Advanced Simulation (IAS-6), Jülich Research Centre, Germany

LASCON X — 9 Jan 2026

Co-funded by
the European Union

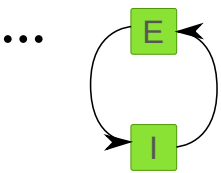
Acknowledgments

This presentation is based on previous work by many people:

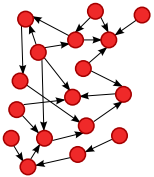
- Hannah Bos
- David Dahmen
- Moritz Deger
- Jochen Martin Eppler
- Espen Hagen
- Abigail Morrison
- Jannis Schuecker
- Johanna Senk
- Tom Tetzlaff
- Sacha van Albada
- Charl Linssen

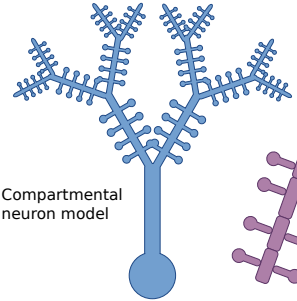
When to use NEST?

Population model



Point neuron network model





Compartmental neuron model

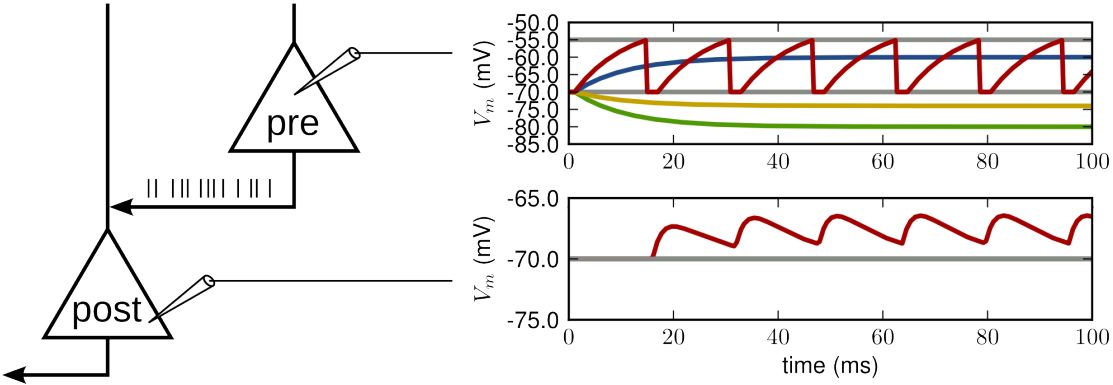
Compartmental membrane model

Reaction-diffusion model

Complexity of single elements

Possibility to simulate large networks

NEST simulation mimics a neuroscientific experiment



NEST = NEural Simulation Tool

- Simulator for spiking neural networks with a focus on the dynamics, size and structure of neural systems, rather than on the exact morphology of individual neurons

- C++ core, hybrid parallelization (OpenMP+MPI), Python frontend PyNEST
- Same code from laptops to supercomputers $\rightarrow$ simulation of large-scale models

- Development is driven by scientific needs with a focus on accuracy and flexibility as well as quality assurance

https://www.nest-simulator.org
https://github.com/nest/nest-simulator
https://nest-simulator.readthedocs.io
https://ebrains.eu/data-tools-services/tools/nest

Co-funded by the European Union

Built-in neuron models

- Integrate-and-fire models (`iaf_*`)
  - Current-based (`iaf_psc_*`)
  - Conductance-based (`iaf_cond_*`)
  - Different post-synaptic shapes (`*_delta`, `*_alpha`, `*_exp`)
  - Width precise spike timing (not on grid, `*_ps`)
- Adaptive exponential integrate-and-fire models (`aeif_*`)
- Allen Institute generalised leaky integrate-and fire family (`gif_*`)
- Hodgin-Huxley models (`hh_*`)
- Hill-Tononi model **(Hill & Tononi, 2005)** (`ht_*`)
- Astrocytes (`astrocyte_lr_1994`, since NEST 3.6)
- Support for eprop (back-propagation through time, since NEST 3.7)
- Compartmental neurons (`cm_*`)

. . . and many more!

Built-in synapse models

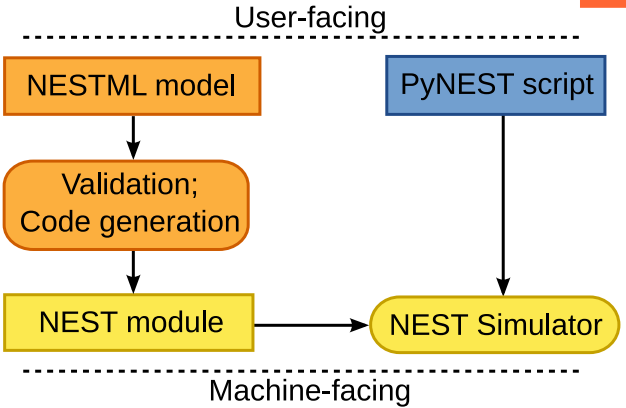
- Static synapse
- Short term depression and facilitation **(Tsodyks, Uziel, Markram, 2000)**
- Spike-timing dependent plasticity (STDP)
  - All-to-all and several nearest-neighbour variants
- Triplet STDP **(Pfister & Gerstner, 2006)**
- Voltage-based STDP **(Clopath et al., 2010)**
- Dopamine-modulated STDP **(Potjans et al., 2010)**
- Gap junctions (electrical synapses)

. . . and many more!

Co-funded by the European Union

Custom models with NESTML

A domain-specific language for neuron and synapse models



https://github.com/nest/nestml

https://nestml.readthedocs.io

Stimulation devices

Spike generators

- `spike_train_injector` spikes at prescribed points in time
- `poisson_generator` spikes according to a Poisson distribution
- `sinusiodal_poisson,gamma_generator` spikes according to a sinusiodally modulated Poisson or Gamma distribution

Current generators

- `dc_generator` provices a constant current
- `ac_generator` provides a sine-shaped current
- `step_current_generator` provides a step-wise constant current
- `noise_generator` provides a random noise current

Co-funded by the European Union

Recording devices

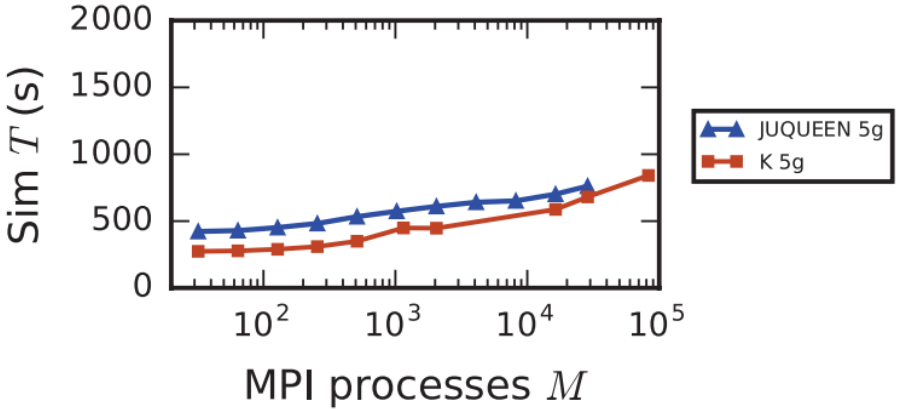
- `spike_recorder` records incoming spikes
- `multimeter` records analog quantities (potentials, conductances, ...)
- `voltmeter` records the membrane potential
- `correlation_detector` records pairwise cross-correlations between the spiking activity of neurons
- `weight_recorder` records the weight of connections

Parameterization

Parameters can be used to set node status, to create positions in space, and to define connection probabilities, weights, and delays.

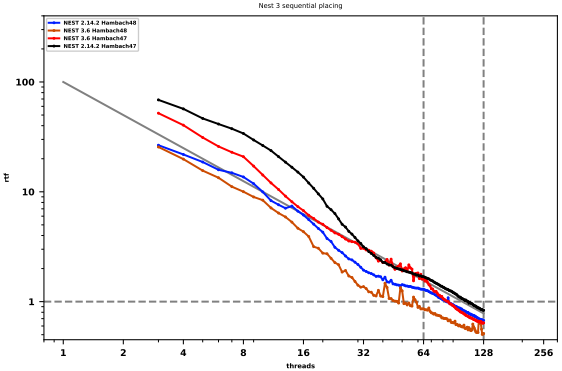
- Random parameters
- Spatial parameters
- Spatially distributed parameters
- Mathematical functions
- Clipping, redrawing, and conditional parameters
- Combination of parameters

Performance: Weak scaling



Jordan, Ippen, Helias, Kitayama, Sato, Igarashi, Diesmann & Kunkel (2018) Front. Neuroinform.

Performance: Strong scaling



Potjans-Diesmann Microcircuit ($\sim$ 80.000 neurons) on 2$\times$ AMD Epyc Rome 64 cores
(Sebastian Gillesen)

NEST GPU

- Previously developed as NeuronGPU, recently joined the NEST Initiative
- Ongoing work:
  alignment to the community-centered workflow of NEST:
  - Documentation
  - CMake installation
  - PyNEST interface
  - NESTML
  - Continuous integration
  - Tests
  - Take NEST GPU to JUPITER

https://github.com/nest/nest-gpu

Golosio, Tiddia, De Luca, Pastorelli, Simula & Paolucci (2021) Front. Comput. Neurosci.

Tiddia, Golosio, Albers, Senk, Simula, Pronold, Fanti, Pastorelli, Paolucci & van Albada (2022) Front. Neuroinform.

Golosio, Villamar, Tiddia, Pastorelli, Stapmanns, Fanti, Paolucci, Morrisson & Senk (in preparation)

Large community with over 30 years of experience

SYNOD: An Environment for Neural Systems Simulations

Language Interface and Tutorial

Markus Diesmann[1], Marc-Oliver Gewaltig[2], and Ad Aertsen[1]

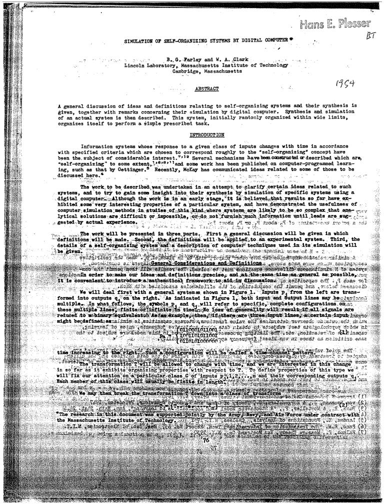
```
r9 | gewaltig | 1996-06-23 11:26:44 +0200 (Sun, 23 Jun 1996) | 2 lines

Initial revision
```

- Online documentation
- User Mailing List
- Open Development on Github
- Bi-weekly developer VC
- Annual virtual conference

Co-funded by the European Union

Ideas for NEST-related projects

- Gallery of NEST Examples
- Papers based on NEST Simulations
- Reproduce Farley & Clark (1954)
- Reproduce Davis *et al* (2021) *Spontaneous traveling waves naturally emerge . . .*
- Explore models with NMDA dynamics using NEST's super-fast approximate implementation
- Explore astrocyte dynamics using NEST astrocyte support (talk to Marja-Leena Linne about the biology!)
- Explore spike-based learning using eligibility propagation support in NEST (more on Tuesday)



Hans E. Plesser

BT

SIMULATION OF SELF-ORGANIZING SYSTEMS BY DIGITAL COMPUTER*

B. G. Farley and W. A. Clark
Lincoln Laboratory, Massachusetts Institute of Technology
Cambridge, Massachusetts

1954

ABSTRACT

A general discussion of ideas and definitions relating to self-organizing systems and their synthesis is given, together with remarks concerning their simulation by digital computer. Synthesis and simulation of an actual system is then described. This system, initially randomly organized within wide limits, organizes itself to perform a simple prescribed task.

INTRODUCTION

Information systems whose response to a given class of inputs changes with time in accordance with specified criteria which are chosen to correspond roughly to the "self-organizing" concept have been the subject of considerable interest.[7,12] Several mechanisms have been constructed or described which are, "self-organizing" to some extent,[1,6,8,11] and some work has been published on computer-programmed learning, such as that by Oettinger.[5] Recently, McKay has communicated ideas related to some of those to be discussed here.[4]

The work to be described was undertaken in an attempt to clarify certain ideas related to such systems, and to try to gain some insight into their synthesis by simulation of specific systems using a digital computer. Although the work is in an early stage, it is believed that results so far have exhibited some very interesting properties of a particular system, and have demonstrated the usefulness of computer simulation methods in studies of this kind where systems are likely to be so complex that analytical solutions are difficult or impossible, or do not furnish much information until leads are suggested by actual experience.

The work will be presented in three parts. First a general discussion will be given in which definitions will be made. Second, the definitions will be applied to an experimental system. Third, the details of a self-organizing system and a description of computer techniques used in its simulation will be given.

General Considerations and Definitions

In order to make our ideas and definitions precise, and at the same time as general as possible, it is convenient to introduce a mathematical framework to aid in discussion.

We will deal first with a general system as shown in Figure 1. Inputs p_j from the left are transformed into outputs q_j on the right. As indicated in Figure 1, both input and output lines may be multiple. In what follows, the symbols p_j and q_j will refer to specific, complete configurations on these multiple lines, finite or infinite in time. No loss of generality will result if all signals are reduced to a binary equivalent. As an example, then, if there are three input lines, a certain input might be defined as

$$p_a = \begin{cases} 011000010101001 \\ 100110101011001 \\ 011010100000000 \end{cases} \quad (1)$$

time increasing to the right. Such a configuration will be called a time-channel pattern.

The transformation T will be allowed to change with time, and we are interested in this change in so far as it exhibits organizing properties with respect to T. To define properties of this type we will fix our attention on a particular class C of inputs p_j, j=1,2,...,n and their corresponding outputs q_j. Each member of this class will usually be finite in length.

We may then break the transformation T down into a class of transforms

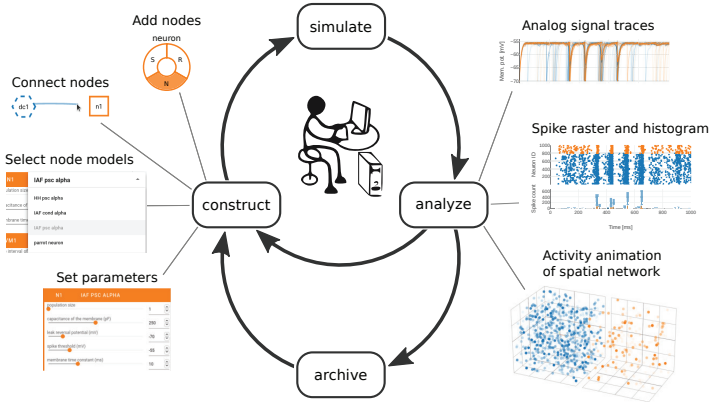
$$T = \{T_1, T_2, \ldots, T_n\} \quad (2)$$

*The research in this document was supported jointly by the Army, Navy, and Air Force under contract with the Massachusetts Institute of Technology.

76

NEST Desktop—a GUI for NEST

- Web-based
  graphical user interface
  as educational application
- Interactive construction and
  exploration of neural networks



Spreizer, Senk, Rotter, Diesmann & Weyers (2021) eNeuro

Working with NEST Desktop

Documentation https://nest-desktop.readthedocs.io

On EBRAINS No installation required

- Register on https://ebrains.eu/sign-up (use **institutional** email)
- Use NEST Desktop at https://ebrains.eu/service/nest-desktop

On your computer Using the VM or the Conda/Mamba setup

- In `Terminal`, run

```
export NEST_SERVER_DISABLE_AUTH=1
nest-server start -d
nest-desktop start
```

- Open the URL shown after `nest-desktop start` in your browser
- To shut down, first use `Ctrl-C` to stop `nest-desktop`, then

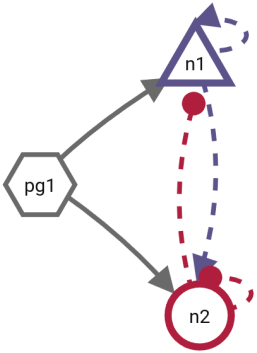
```
nest-server stop
```

- We explore examples provided with NEST Desktop

A balanced random network in NEST Desktop

Recurrently connected networks of excitatory and inhibitory neuron populations with Poissonian input

See also `LasconX_NESTDesktop_Brunel.json`.



Brunel (2000) J Comput Neurosci

Try it out without installation in the browser:
https://nest-desktop.apps.hbp.eu

- `iaf_psc_delta`
- $N_E = 800$, $N_I = 200$
- $C_E = 80$, $C_I = 20$, $D = 1.5\text{ms}$
- $g = 5$, $J_E = J = 1$, $J_I = -gJ = -5$
- $\nu_{\text{ext}} = 1\,000\text{s}^{-1}$
- $C_m = 1\text{pF}$, $\tau_m = 20\text{ms}$
- $E_L = V_{\text{reset}} = 0\text{mV}$, $V_{th} = 20\text{mV}$
- $t_{\text{ref}} = 2\text{ms}$

Human Brain Project

EBRAINS

Thank you

www.humanbrainproject.eu

www.ebrains.eu

Co-funded by
the European Union