Simulating Neuronal Networks with PyNEST

It is easy to do – but do we know what we are doing?

Hans Ekkehard Plesser
UMB / Simula Research Center

Overview

- NEST Initiative & Simulator
  - PyNEST: Simple examples
  - PyNEST: Network examples
  - “Showtime”
- How “science” are simulations today?
  - Reproducibility
  - Sharing and re-use
  - Perspectives

NEST

Initiative & Simulator

The NEST Initiative

- Goals of the NEST Initiative:
  1. share expertise
  2. combine resources
  3. coordinate research and software development
  4. increase software quality
  5. share simulations and results

- Neural Simulation Technology (NEST) Initiative was established in 2001.
- Currently four core developing groups:
  1. Honda Research Institute Europe
  2. BCCN Freiburg
  3. Norwegian University of Life Science, Ås
  4. Brain Science Institute, RIKEN

- Activities:
  - http://www.nest-initiative.org
  - Public releases of NEST available since fall 2004
  - Since 2004 used at the EU Advanced Course in Computational Neuroscience
  - NEST Workshops at the CNS*2005 in Madison (USA) and CNS*2006 in Edinburgh (UK)

Computational Neuroscience

"The goal of neural modeling is to relate, in nervous systems, function to structure on the basis of operation."

MacGregor & Lewis 1977

Important properties

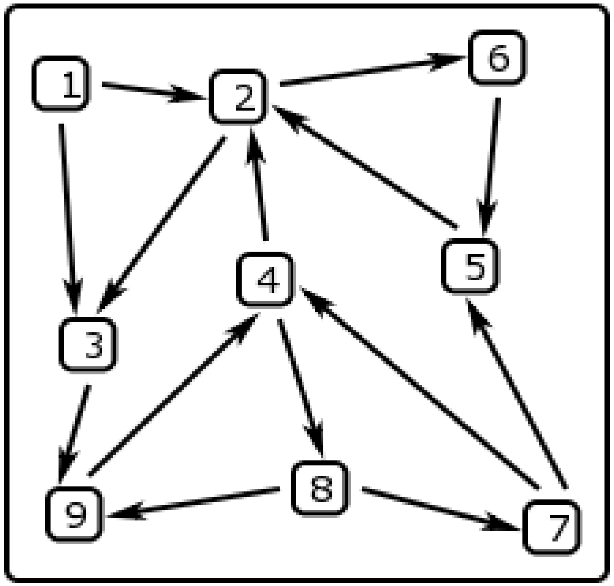
- Discrete processing units: *neurons*
- Neurons process incoming signals internally
- Neurons communicate through stereotypical pulses with finite transmission delays: *spikes*
- Typical scales
  - internal dynamics ~ 1ms
  - spike delays ~ 1ms
  - firing rates ~ 10 Hz
  - 10^5 neurons/mm^3
  - 10^4 inputs/output per neuron

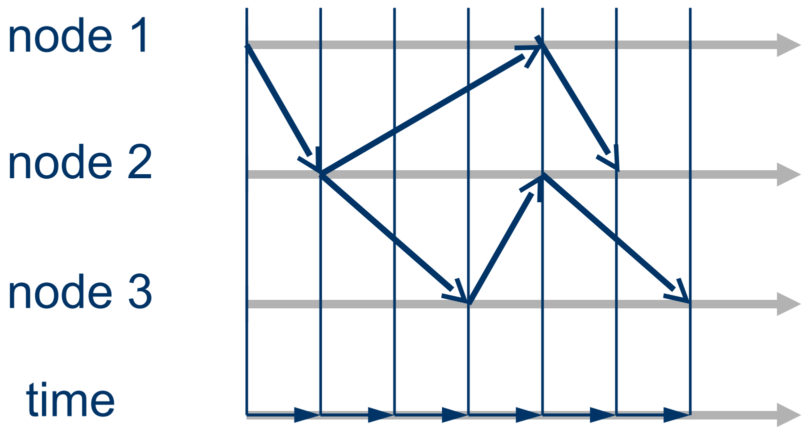
NEST in a nutshell

- Available from www.nest-initiative.org
- Command-line application
- Network models built from neurons, synapses, and devices.
- High-level simulation language (Python or SLI).
- Models for neurons, synapses, and devices are written in C++
- Support for parallel and distributed simulation (Threads and MPI)
- Used at international summer schools since 2004
- «Tell us and cite us» open source license

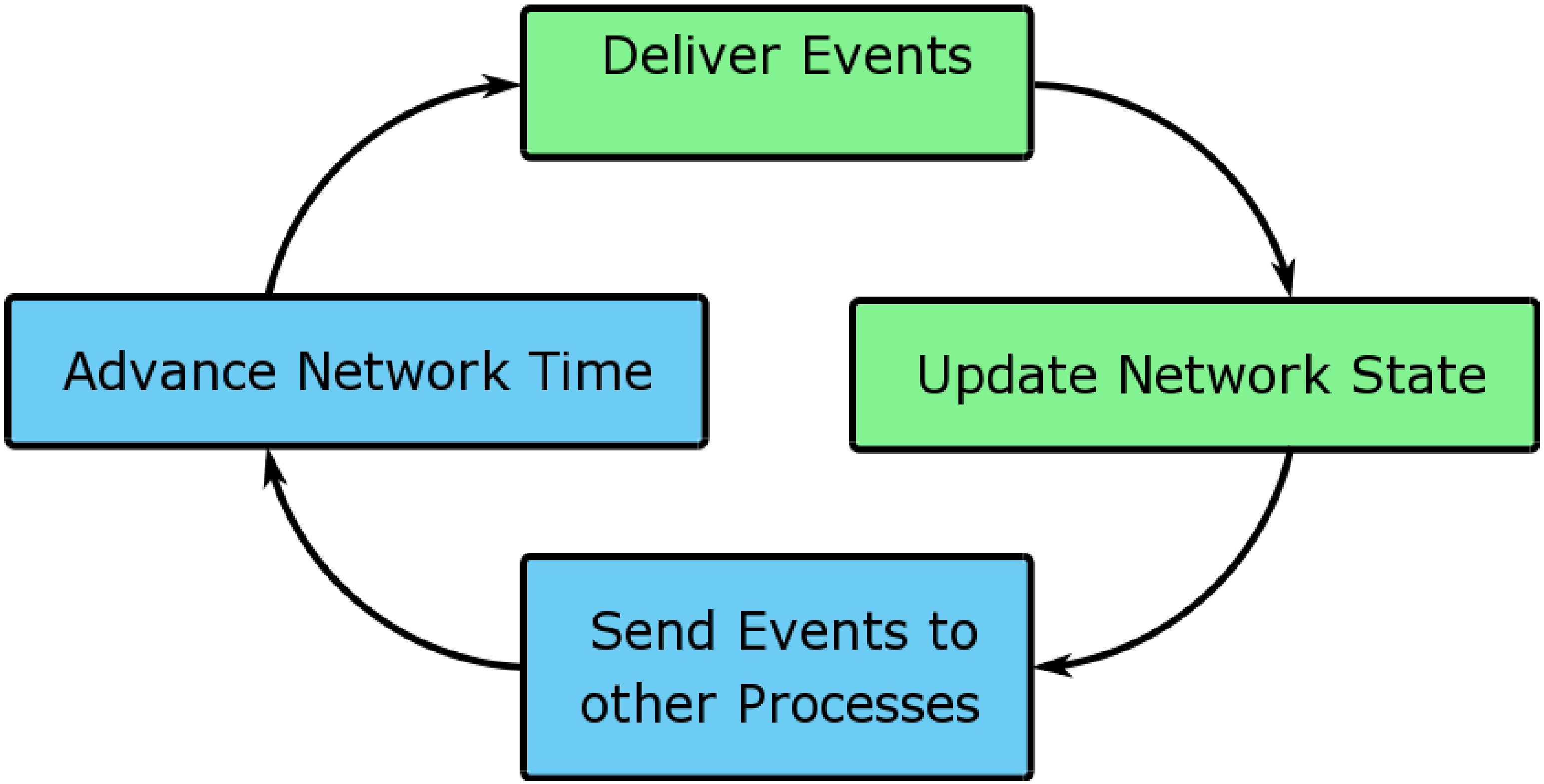
Neural Networks in NEST

- Network: Directed graph
- Nodes: Neurons (10^4-10^6)
- Edges: Connections (10^7-10^9)
- Interaction: **Delayed** pulses
- Kernel tasks:
  - Process incoming spikes
  - Advance neuron states
  - Emit outgoing spikes





Parallel Simulation in NEST2



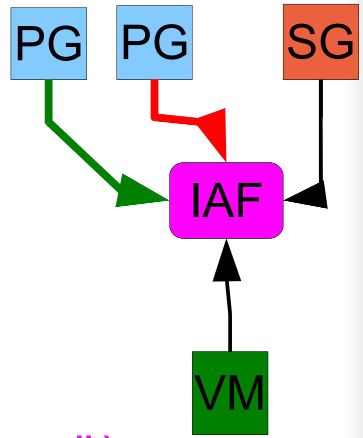
SERIAL

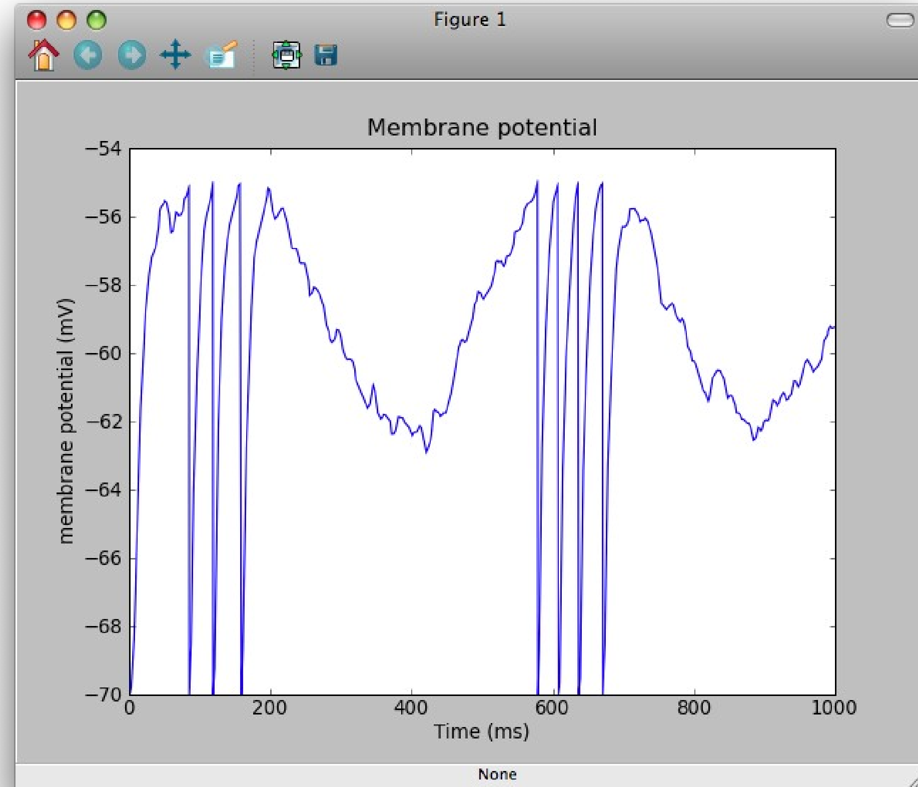
PARALLEL

NEST

Simple Examples

A simple model

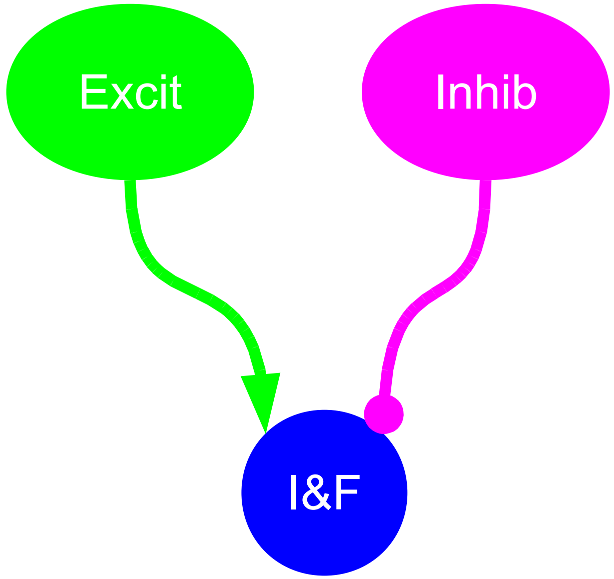





```
import nest
import nest.voltage_trace

nest.ResetKernel()

neuron=nest.Create("iaf_neuron")
noise =nest.Create("poisson_generator",2)
sine  =nest.Create("ac_generator")
voltmeter= nest.Create("voltmeter")

nest.SetStatus(noise,[{"rate":75000.0},
                      {"rate":20000.0}])
nest.SetStatus(sine, [{"amplitude":100.0,
                       "frequency":2.0}])

nest.SetStatus(voltmeter,[{"withgid": True, "withtime": True}])

nest.ConvergentConnect(noise,neuron,weight=[1., -1.])
nest.Connect(voltmeter,neuron)
nest.Connect(sine,neuron)

nest.Simulate(1000.0)

nest.voltage_trace.from_device(voltmeter)
```

Example: Optimizing a network

- Excitatory population modeled as Poisson process
- Inhibitory population modeled as Poisson process
- Single I&F neuron receiving input from both populations
- Goal: Adjust inhibitory population rate so neuron fires with same rate as excitatory population
- Approach: repeated simulation + bisection

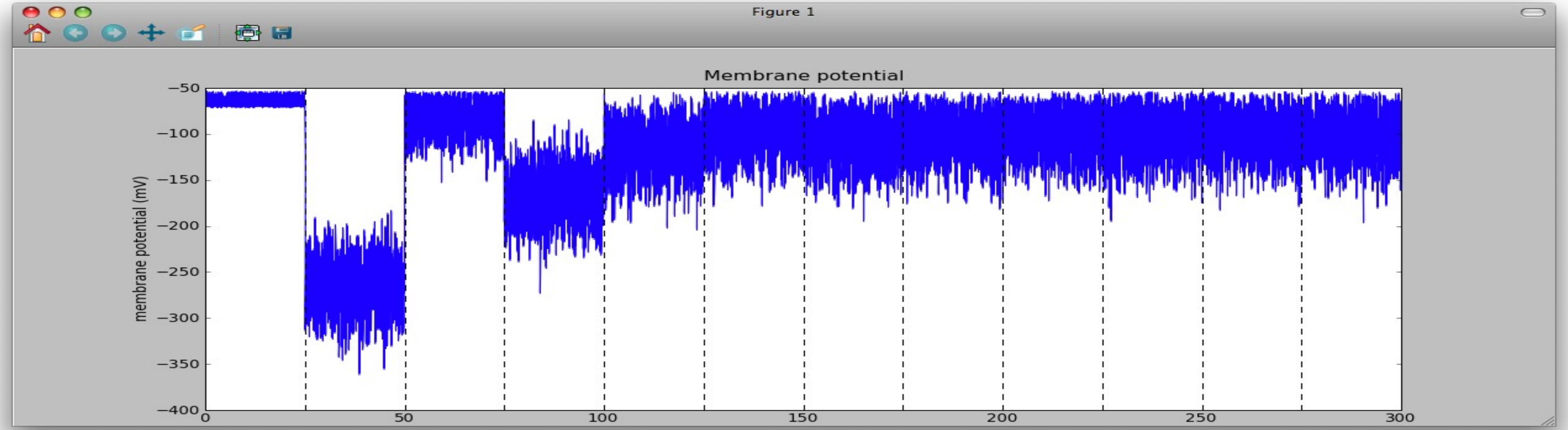


Optimizing: the code

```
neuron         = nest.Create("iaf_neuron")
noise          = nest.Create("poisson_generator",2)
spikedetector = nest.Create("spike_detector")

nest.SetStatus(noise, [{"rate": n_ex*r_ex}, {"rate": n_in*r_in}])

nest.ConvergentConnect(noise, neuron, [w_excit, w_inhib], 1.0)
nest.Connect(neuron, spikedetector)

in_rate = bisect(lambda x: output_rate(x) - r_ex,
                 lower, upper, xtol=prec)


def output_rate(guess):
    rate = float(abs(n_in*guess))
    nest.SetStatus([noise[1]], "rate", rate)
    nest.SetStatus(spikedetector, "n_events", 0)
    nest.Simulate(t_sim)
    out=nest.GetStatus(spikedetector, "n_events")[0]*1000.0/t_sim
    return out
```

Optimization: example run

```
Inhibitory rate estimate: 15.00 Hz -> Neuron rate: 347.64 Hz
Inhibitory rate estimate: 25.00 Hz -> Neuron rate:   0.04 Hz
Inhibitory rate estimate: 20.00 Hz -> Neuron rate:  37.04 Hz
Inhibitory rate estimate: 22.50 Hz -> Neuron rate:   0.00 Hz
Inhibitory rate estimate: 21.25 Hz -> Neuron rate:   0.92 Hz
Inhibitory rate estimate: 20.62 Hz -> Neuron rate:   7.32 Hz
Inhibitory rate estimate: 20.94 Hz -> Neuron rate:   3.48 Hz
Inhibitory rate estimate: 20.78 Hz -> Neuron rate:   3.92 Hz
Inhibitory rate estimate: 20.70 Hz -> Neuron rate:   6.04 Hz
Inhibitory rate estimate: 20.74 Hz -> Neuron rate:   5.76 Hz
Inhibitory rate estimate: 20.76 Hz -> Neuron rate:   5.24 Hz
Inhibitory rate estimate: 20.77 Hz -> Neuron rate:   5.28 Hz

Optimal rate for the inhibitory population: 20.77 Hz
```



PyNEST

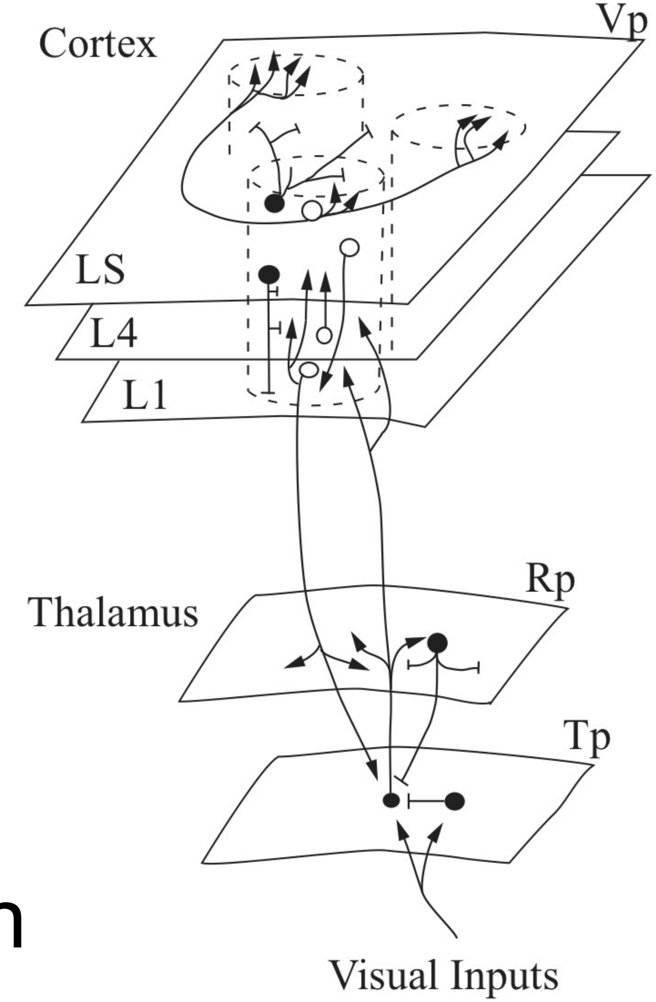
Network Example

NEST Topology Module

- Idea: User-friendly support for layered networks
- Implementation: Kittel Austvoll
- Describe network as collection of layers
- Elements of a layer can be
  - Individual neurons
  - Groups of neurons (e.g. Microcolumn)
  - Placed on a fixed grid
  - Placed arbitrarily in space
- Connections described by
  - Masks: no connections outside mask
  - Kernels: give distance-dependent connection probability

Example Network
(after Lumer et al, 1997)

- LGN: Layer of individual neurons
- Vp: Layer of microcolums
  - L4: 2 pyr. cells, 1 internrn
  - L6: 1 pyramidal cell
- Connections:
  - LGN -> Vp/L4 pyr: rectangle
  - Vp/L4 pyr -> internrn: circular
  - Vp/L4 internrn -> pyr: “doughnut”
  - Vp/L4 internrn -> internrn: “flat”
  - Vp/L4 pyr -> Vp/L6 pyr: Gaussian
  - Vp/L6 pyr -> LGN: Gaussian



After Lumer et al, 1997

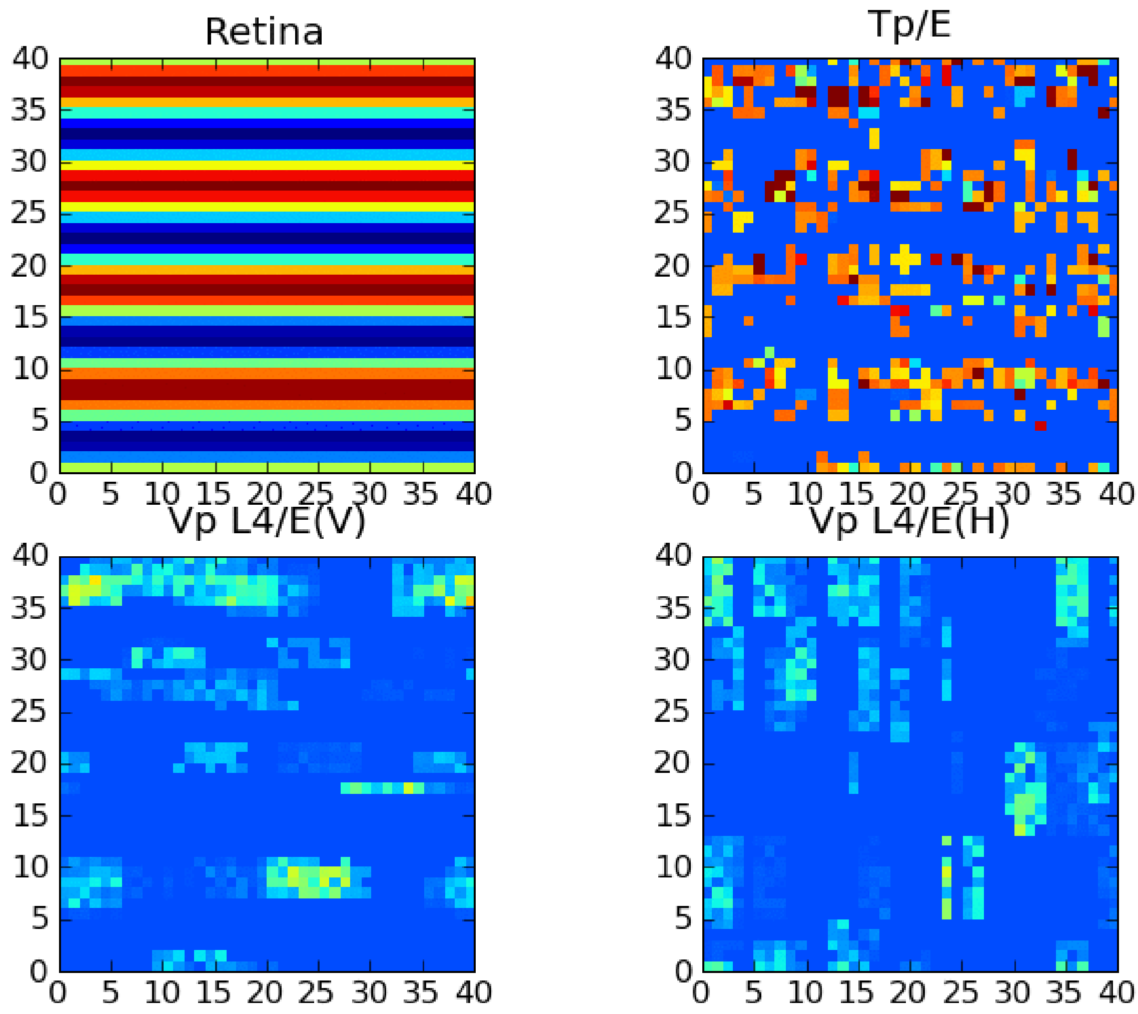
PyNEST/Topology Code for Network

```
layout = {'rows': 50, 'columns': 50,
          'extent': [5.0, 5.0],
          'center': [0.0, 0.0],
          'edge_wrap': False}

# - Retina --------------------------------

nest.CopyModel('ac_poisson_generator',
               'retina_cell',
               {'DC': 30.0, 'AC': [30.0],
                'Freq': [2.0], 'Phi': [0.0]})

ret_config = dict(layout)
ret_config['elements'] = 'retina_cell'

retina = nest.CreateLayer(ret_config)

# - LGN -----------------------------------

nest.CopyModel('iaf_neuron', 'lgn_rc')

lgn_config = dict(layout)
lgn_config['elements'] = 'lgn_rc'

lgn = nest.CreateLayer(lgn_config)

# - V1 ------------------------------------

nest.CopyModel('iaf_neuron', 'l4_pyr')
nest.CopyModel('iaf_neuron', 'l4_inh')
nest.CopyModel('iaf_neuron', 'l6_pyr')

v1_config = dict(layout)
v1_config['elements']
    = [['l4_pyr', 2, 'l4_inh', 1], ['l6_pyr', 1]]

v1 = nest.CreateLayer(v1_config)
```

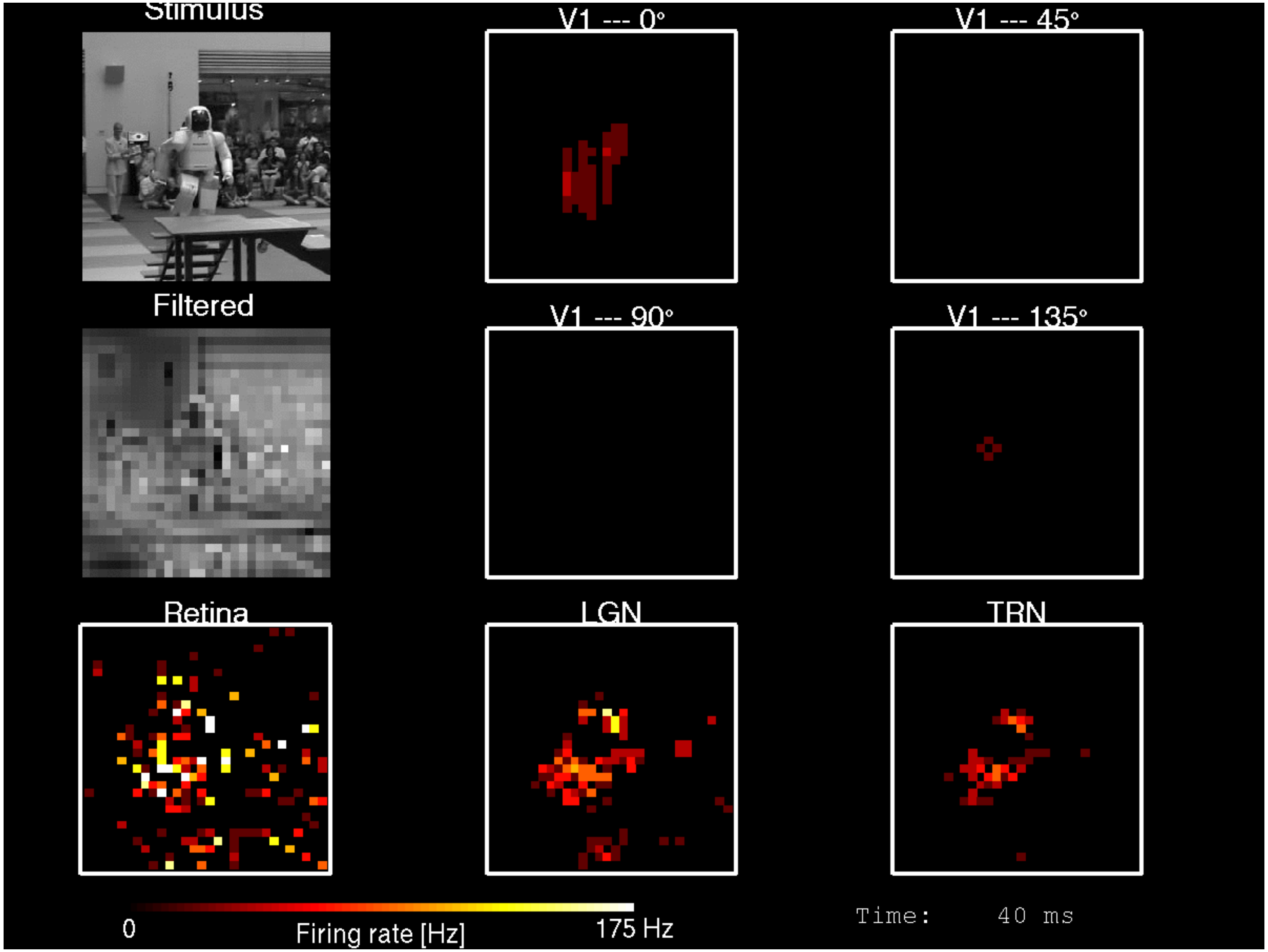
```
# - Retina -> LGN -------------------------

nest.ConnectLayer(retina, lgn,
  {'connection_type': 'convergent',
   'mask': { 'grid': {'rows': 1, 'columns': 1} },
   'delay': 1.0,
   'weight': 10.0})

# - LGN -> V1/L4 --------------------------

nest.ConnectLayer(lgn, v1,
  {'connection_type': 'convergent',
   'targets': {'model': 'l4_pyr'},
   'mask': {'rectangular':
            {'lower_left': [-0.4, -0.1],
             'upper_right': [0.4, 0.1]}},
   'kernel': 0.5,
   'weights': 5.0,
   'delays': {'uniform': {'min': 2, 'max': 3}}})

# - V1/L4 -> V1/L4 ------------------------

nest.ConnectLayer(v1, v1,
  {'connection_type': 'divergent',
   'sources': {'model': 'l4_pyr'},
   'targets': {'model': 'l4_inh'},
   'mask': {'circular': {'radius': 1.0}},
   'kernel': {'linear': {'c': 1.0, 'a': -1.0}},
   'delays': 1.0,
   'weights': 2.0})

# - V1/L4 -> V1/L6 ------------------------
...

# - V1/L6 -> LGN --------------------------
...
```

Showtime

Simplified version of Hill & Tononi (2005)



Larger visual-pathway model



How “science” is neuronal network simulation today?

Scientific method

- Thorough critique of methods, observations, and conclusions
- Validation based on independent reproduction
- Accumulation of knowledge through exchange, evolution and (sometimes) revolution of ideas
- Requires precise and comprehensible description of research

Reproducibility

Neuronal network simulations are generally not reproducible today (and everyone knows ...)

Example 1

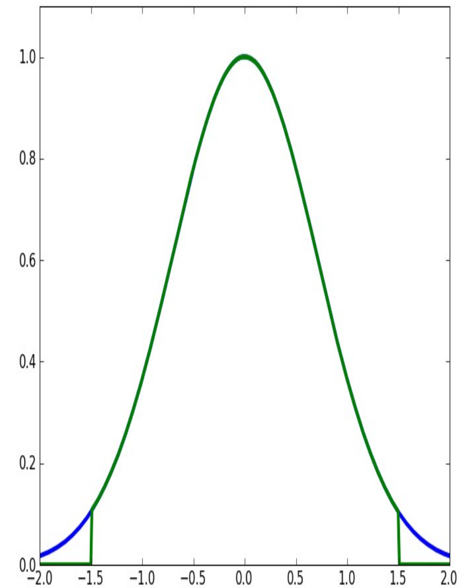
- Single neuron model
- Generally well presented
- Paper-and-pencil analysis shows that row “3” should have no spikes
- Could not be resolved in collaboration with author
- Probably figure mix-up
- No qualitative consequences

Example 2

- Well-known integrate-and-fire network model
- Chosen as benchmark for simulator comparison
- Author of paper unable to reproduce figures from his own paper with his own simulator
- Differences probably due to “minor” changes in simulator code
- Never resolved
- No qualitative consequences

Example 3

- Well-known paper on plasticity
- Neuronal connections based on Gaussian profile
- Reproduction failed qualitatively
- Inspection of original C-code revealed Gaussian with cut-off
- Were original authors aware of role of cut-off?

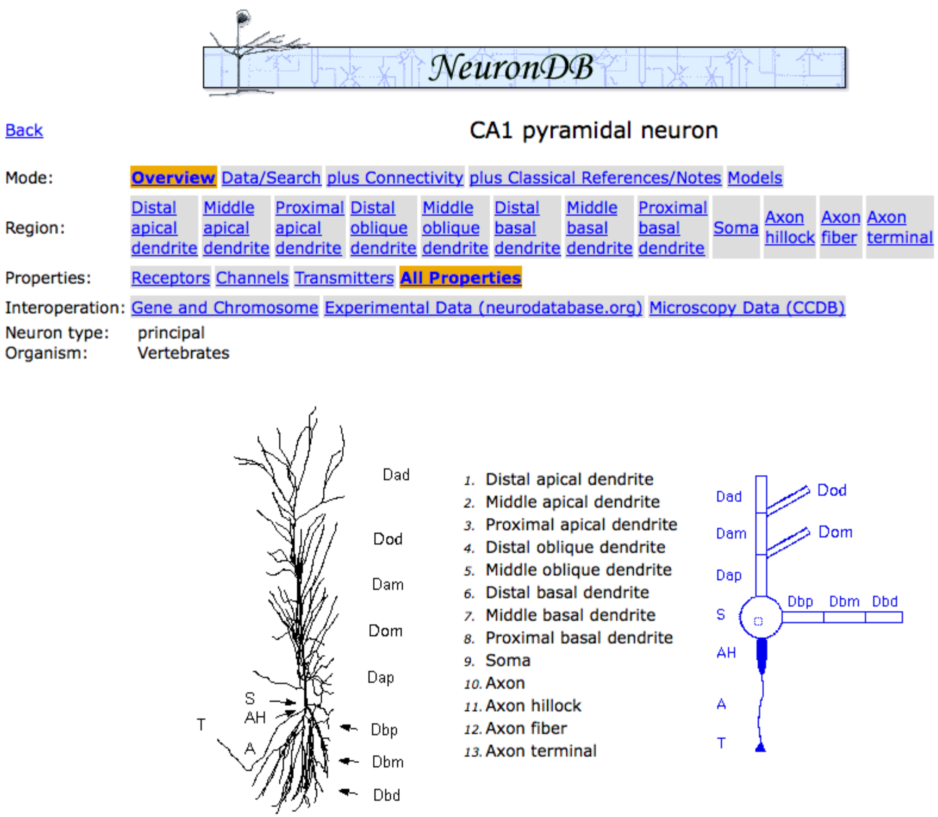


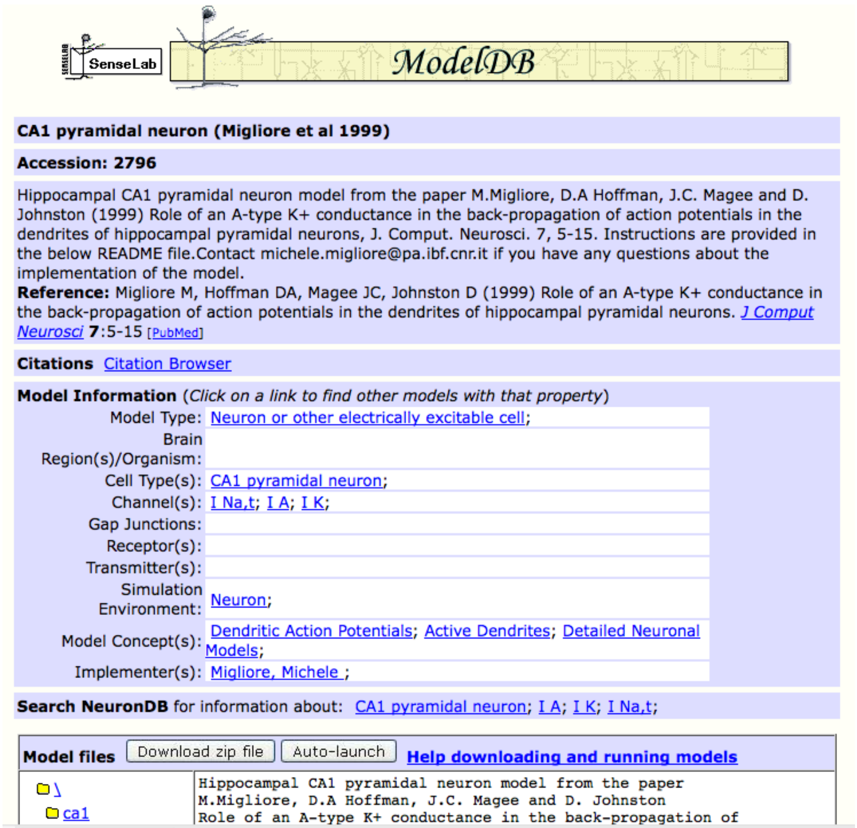
How bad is it?

- “It is currently impossible to reproduce and validate most of the results that computational scientist publish ...” (Stodden, 2009)
- “[A]n article about computational science in a scientific publication is not the scholarship itself, it is merely advertising on scholarship.” (J Claerbout)
- Shooting-star crystallographer had to retract six papers because “a homemade data-analysis program had flipped two columns of data” (Science 314:1856 2006).
- See Victoria Stodden for more (http://www.stanford.edu/~vcs/)

Sharing and re-use

Resources: NeuronDB & ModelDB





But are they used?

- Google Scholar search for “ModelDB Accession Number”:

Analog Modulation of Mossy Fiber Transmission Is Uncoupled from Changes in Presynaptic Ca2+ - ▶jneurosci.org
R Scott, A Ruiz, C Henneberger, DM Kullmann, DA … - Journal of Neuroscience, 2008 - Soc Neuroscience
... dentate granule cell adapted from the study by Schmidt-Hieber et al., 2007 Go)
(cell number 7, imported from SenseLab; **ModelDB accession number** 95960, http ...
Sitert av 1 - Beslektede artikler - Websøk - Alle 2 versjoner

A modeling study suggesting a possible pharmacological target to mitigate the effects of ethanol on …
M Migliore, C Cannia, CC Canavier - Journal of Neurophysiology, 2008 - Am Physiological Soc
Beslektede artikler - Websøk - Alle 4 versjoner

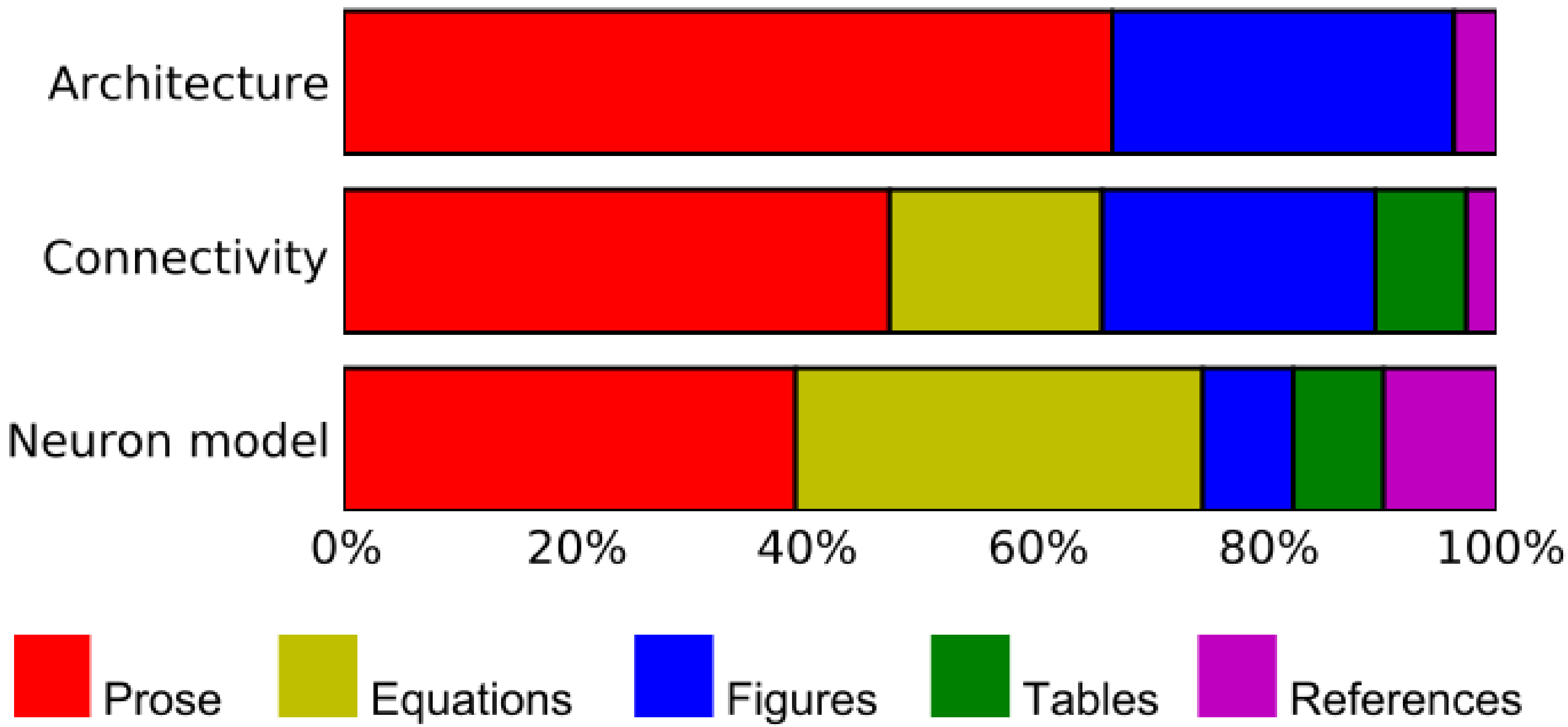
Translating network models to parallel hardware in NEURON - ▶psu.edu [PDF] - BIBSYS eText
ML Hines, NT Carnevale - Journal of Neuroscience Methods, 2008 - Elsevier
Sitert av 2 - Beslektede artikler - Websøk - Alle 3 versjoner

▶Improved Focalization of Electrical Microstimulation Using Microelectrode Arrays: A Modeling Study - BIBSYS eText
S Joucla, B Yvert - PLoS ONE, 2009 - pubmedcentral.nih.gov
... both a straight fiber (Figure 9.A) and a complex CNS neuron, taken from the literature
[32] and obtained from **ModelDB** (**accession number** 2448) (Figure 9.B). The ...
Websøk

Why not?

- Little tradition for use of standard tools/simulators
- Difficult to port models from one simulator to other
- PhD students like to write their own simulators
  - Instant gratification from software development
  - Unaware of pitfalls
  - Desire for “total understanding & control”
  - Lack of compentence among supervisors?
- Models described in widely different ways in literature
- Rarely ever in a way facilitating re-building

Ways of model description



Nordlie, Gewaltig, Plesser (submitted)

Description vs Development

- Models often described with detailed references to biological literature
- Biological justification may obfuscate concise of resulting model architecture
- But in fact
  - Design decisions often based on “what was needed to make model work”
  - Some decisions motivated by external factors, eg need to define student project
- Last to points rarely mentioned in papers
- Information important for re-use lacking

Perspectives

- Increasing awareness of advantages of "standard simulators" (Neuron, NEST, Genesis/ Moose, PCSim, Brian)
- Review by Brette et al, J Comp Neurosci 2007
- Simulator integration (PyNN, Music)
- SBML & CellML showing advantages of standards for sharing models
- INCF Task Force on Standard Language for Neuronal Network Models
- BUT: Requires conscious effort by all in the field