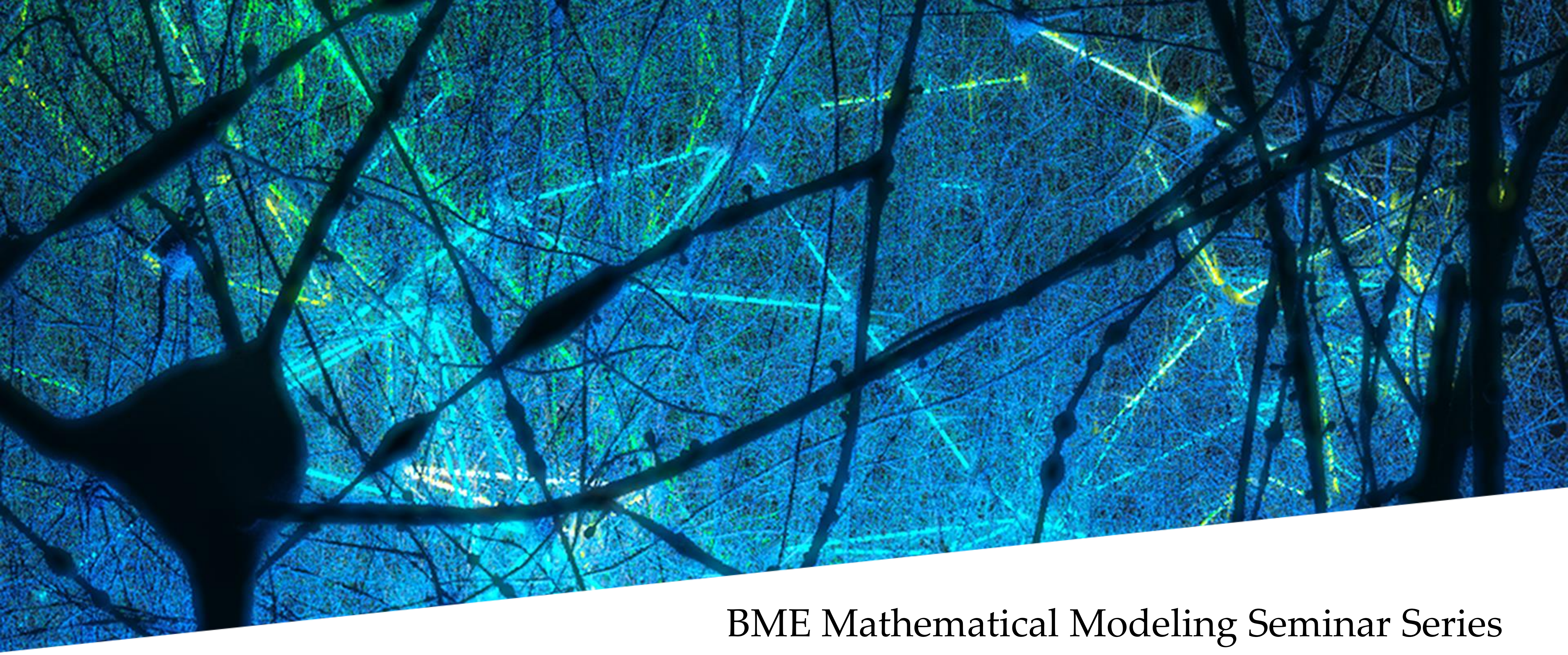


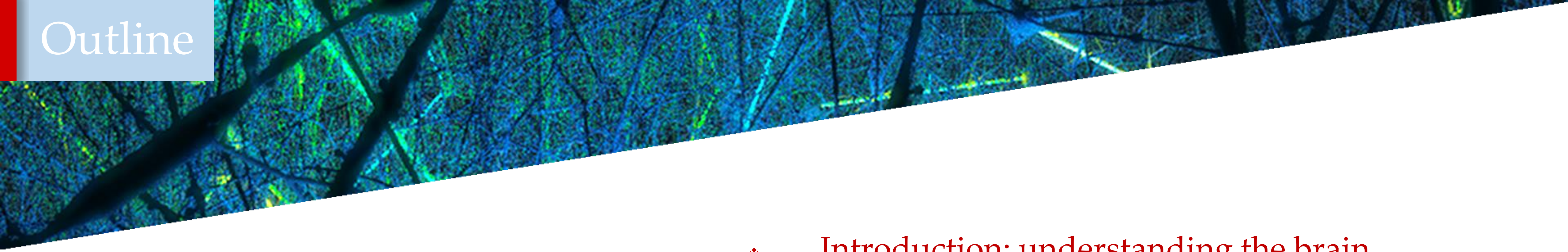
BME Mathematical Modeling Seminar Series

Understanding dynamics and computations in the brain by simulating neurons and networks

Szabolcs Káli

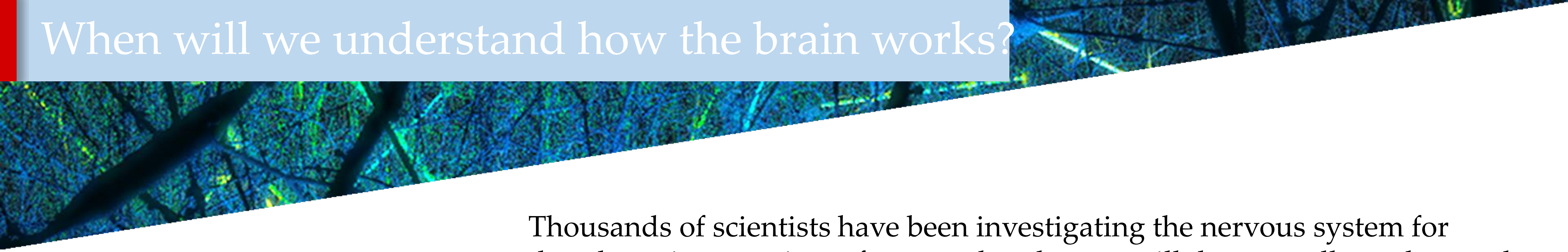
kali@koki.hu

Outline



- Introduction: understanding the brain
- Systematic model construction and evaluation
  - Parameter optimization
  - Automated model validation
- Neural modeling example
  - Model of network dynamics, coding, and learning in the hippocampus

When will we understand how the brain works?



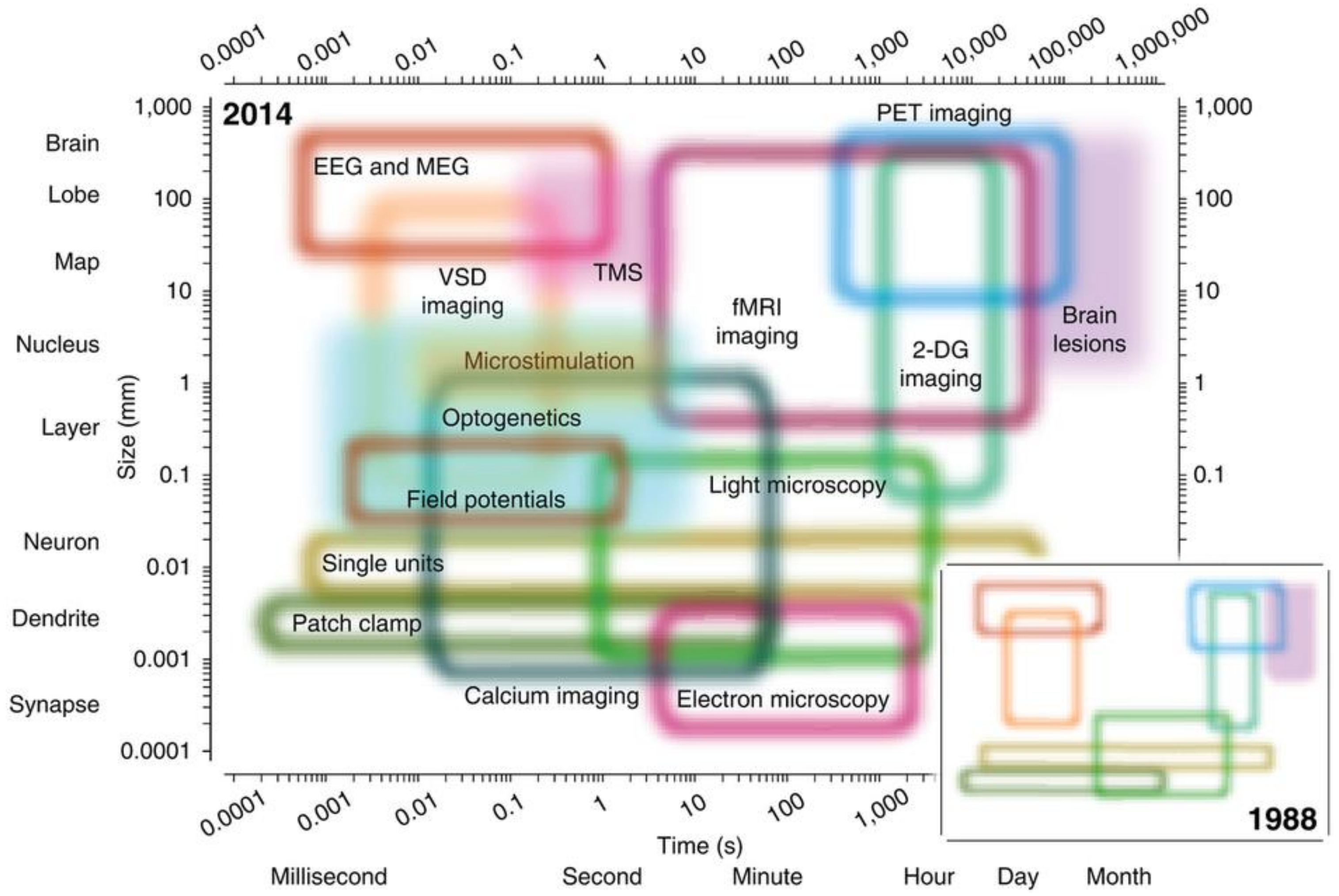
Thousands of scientists have been investigating the nervous system for decades using a variety of approaches, but we still do not really understand how it works. Why?

Here are a few reasons why the problem is very hard:

- A large number of heterogeneous elements with high dimensional, nonlinear internal dynamics
- Many kinds of structured, state-dependent, delayed interactions
- Several different, but overlapping time scales
- Task-specific solutions which appeared through evolution, and form via interactions with the environment during development

Is it then hopeless to understand the brain?
I will argue that it is not…

Levels of organization and methods of investigation



Levels of organization and methods of investigation

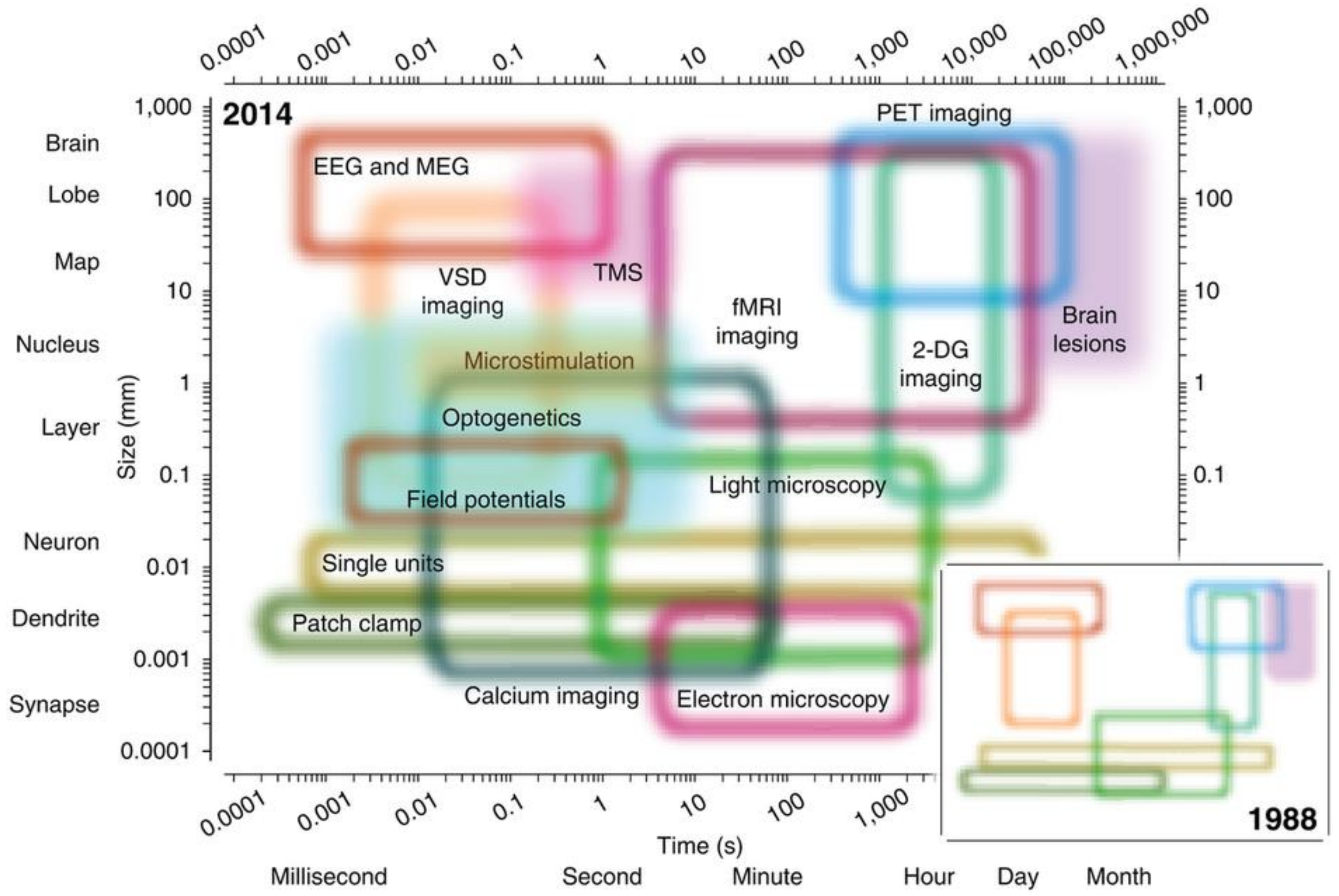
Theories and models can link different levels, but they still capture only a few levels, so they must be tied together somehow:

“bottom-up”

and

“top-down”

strategies



A purely bottom-up strategy

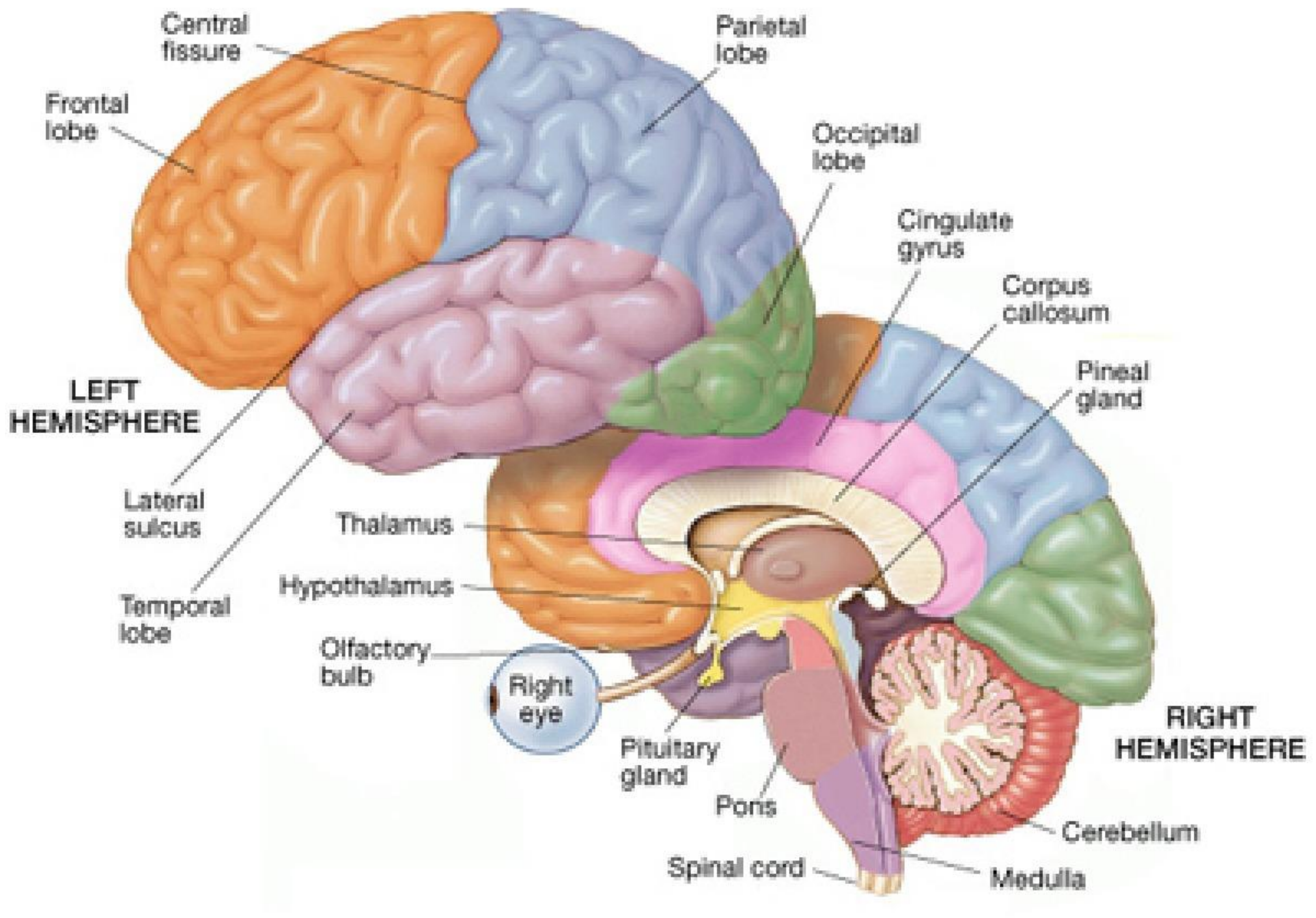
If we modeled all details accurately, we would probably get accurate predictions, but

- it is hopeless to measure all the parameters
- it is impossible to determine the precise initial conditions
- chaotic behavior is also possible

- besides, such a model would have limited utility:
  - it could be used for “in silico” experimentation

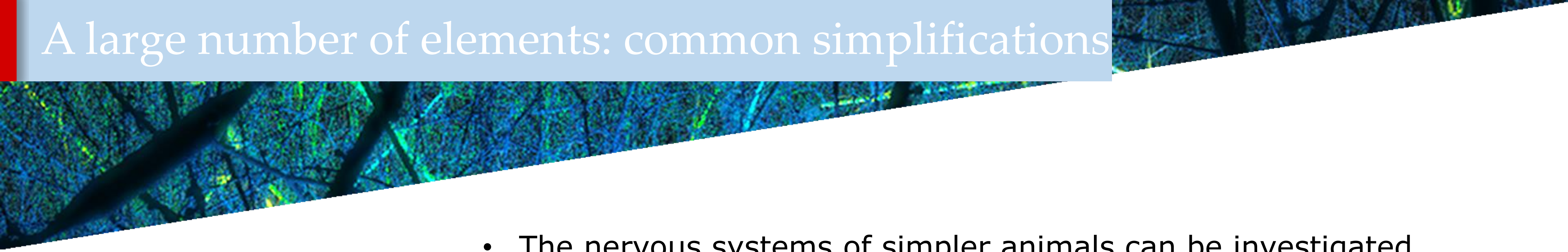
- would such a full model lead to true understanding?
  - depends on the definition of “understanding”…

A large number of elements: regions, neurons, synapses



In the human brain, there are ~10^{11} neurons and ~10^{14} synapses.

A large number of elements: common simplifications



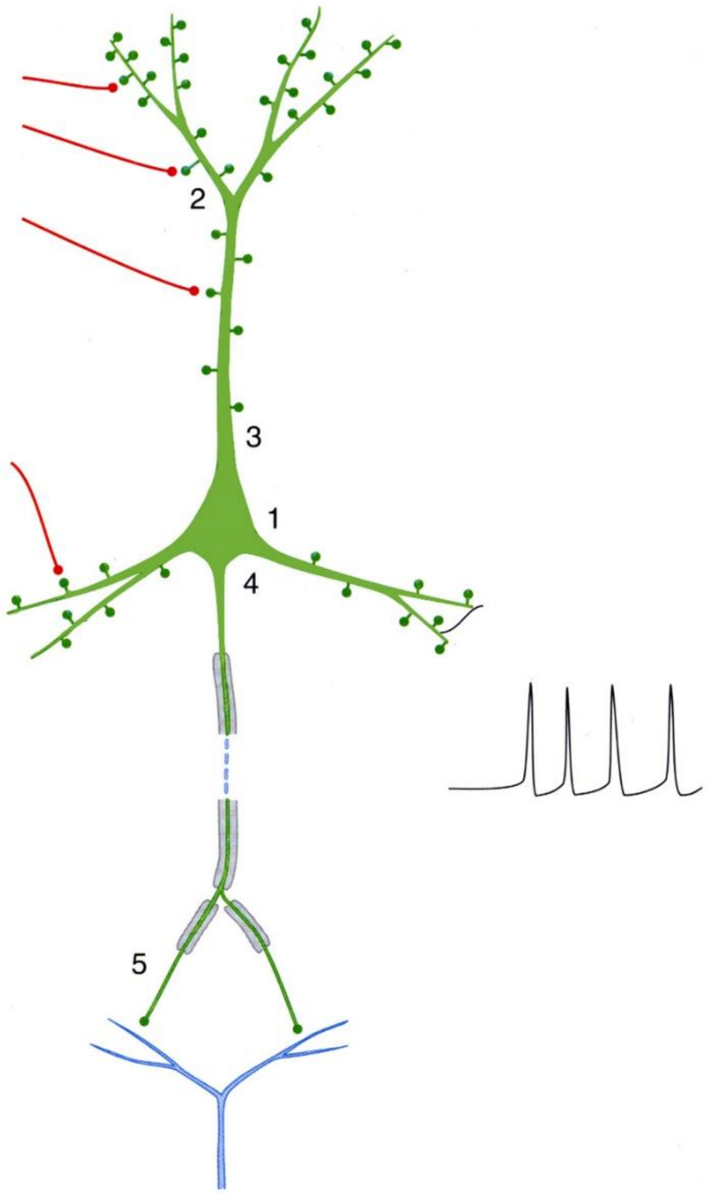
- The nervous systems of simpler animals can be investigated
  - C. elegans (a worm) has 302 neurons, ~6400 chemical and ~900 electric synapses
  - Slugs, insects (Drosophila, locust), fish, etc. are also used

- A single brain region or a few interacting brain regions, or in vitro brain slices can be investigated
  - These contain ~10^4 – 10^9 neurons, ~10^7-10^{12} synapses

- In models: we can use the average activity of neuronal populations as our state variables

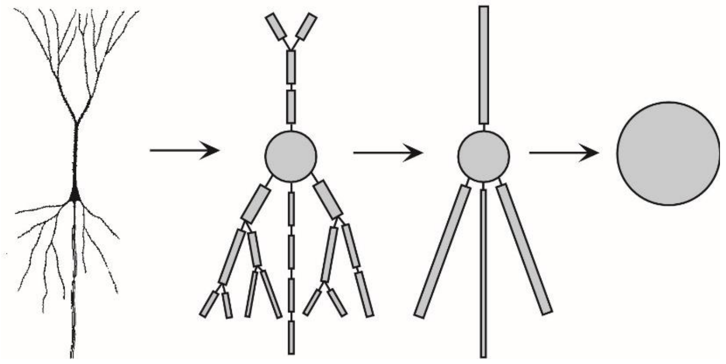
Complex elements: neurons and synapses

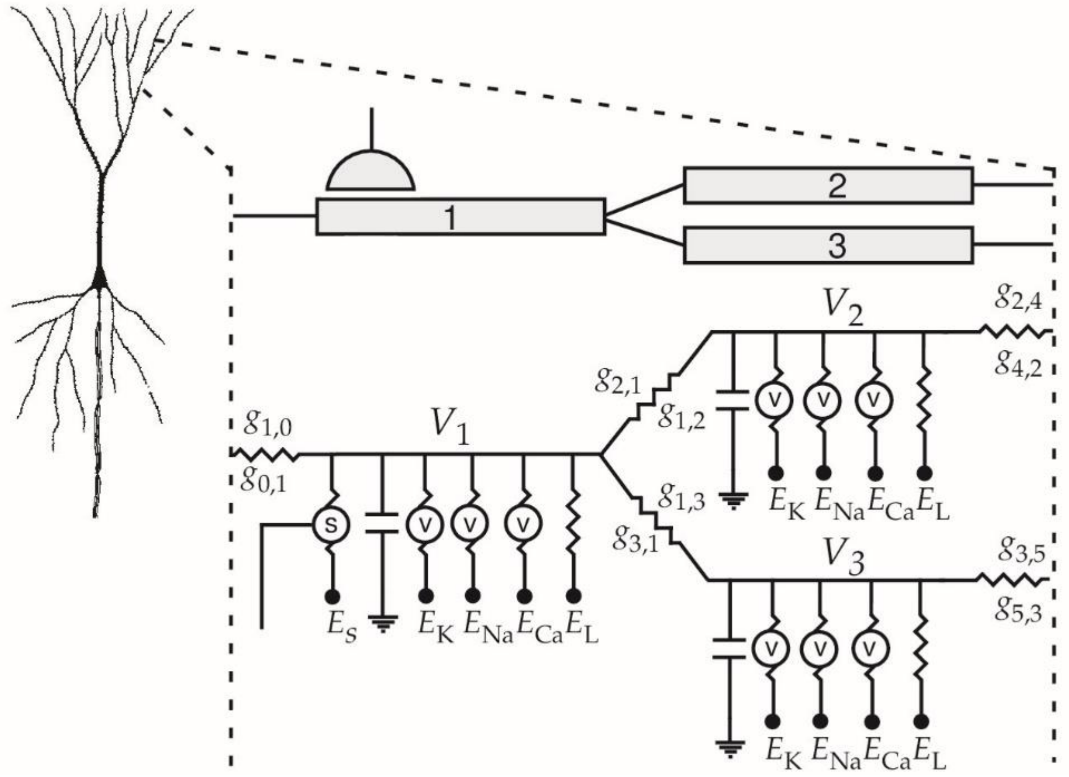
- Electric and chemical signals
- Complex, nonlinear processing within neurons
  - Dendritic action potentials
- Synaptic transmission:
  - Excitatory, inhibitory, modulatory
  - State-dependent, delayed, plastic on several different timescales



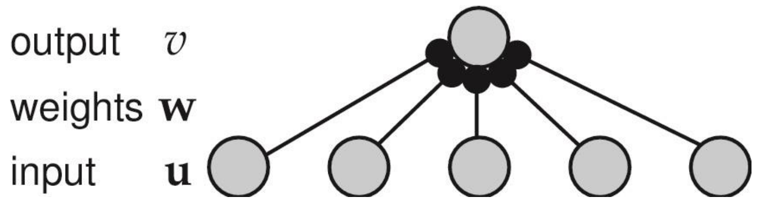
Complex elements: common simplifications

- Morphologically and biophysically detailed, multi-compartmental models -> simplified, spiking or rate-based neuronal models

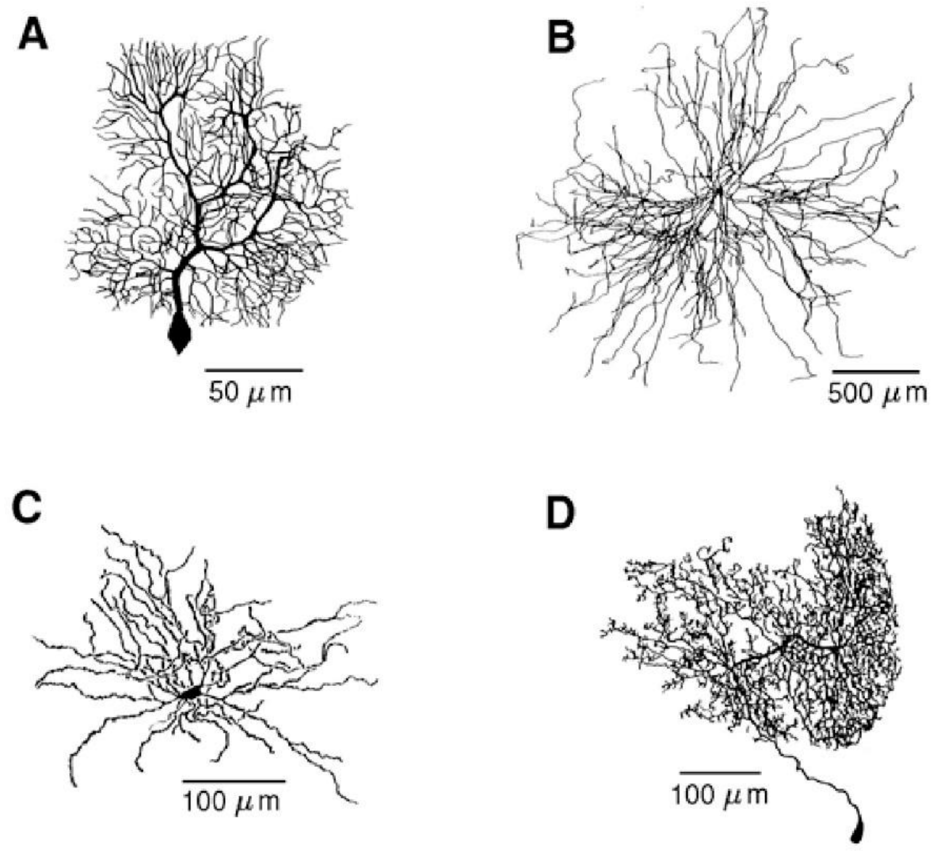


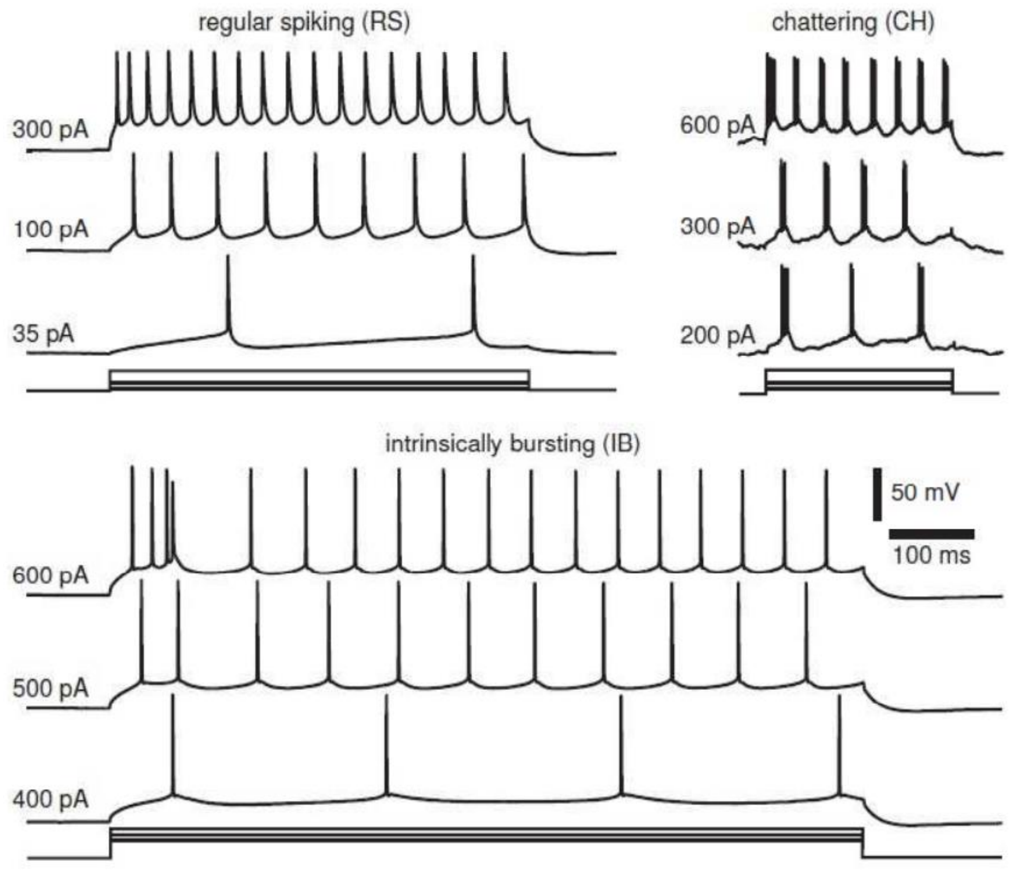


- Synapses are also information-processing devices, which have their own state variables – this is often neglected



Heterogeneous elements

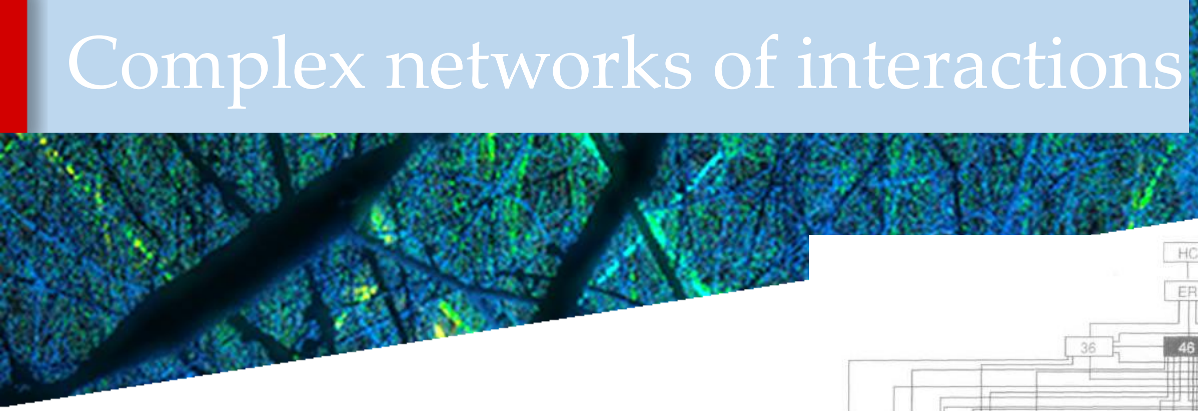


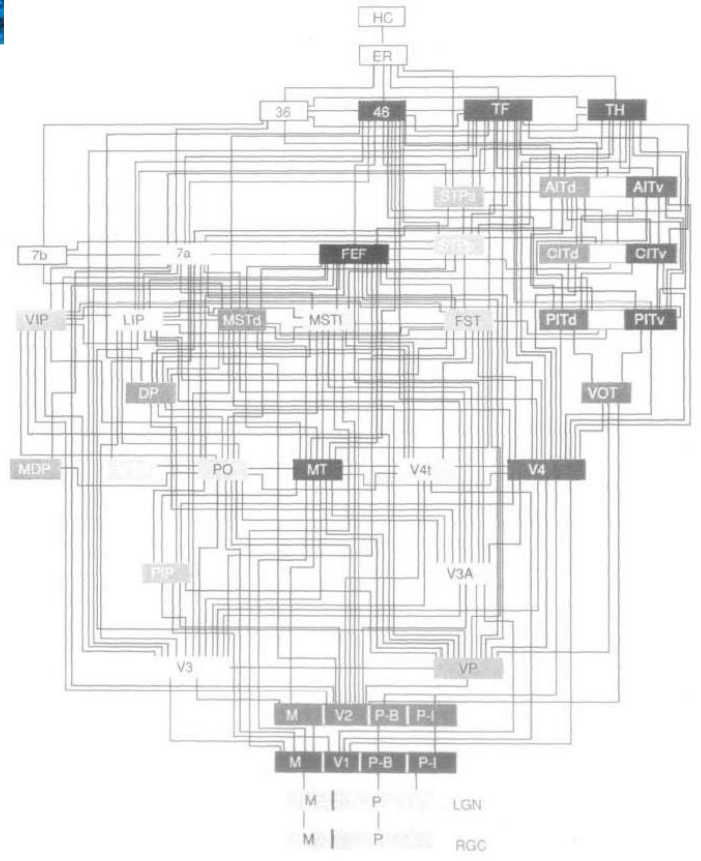


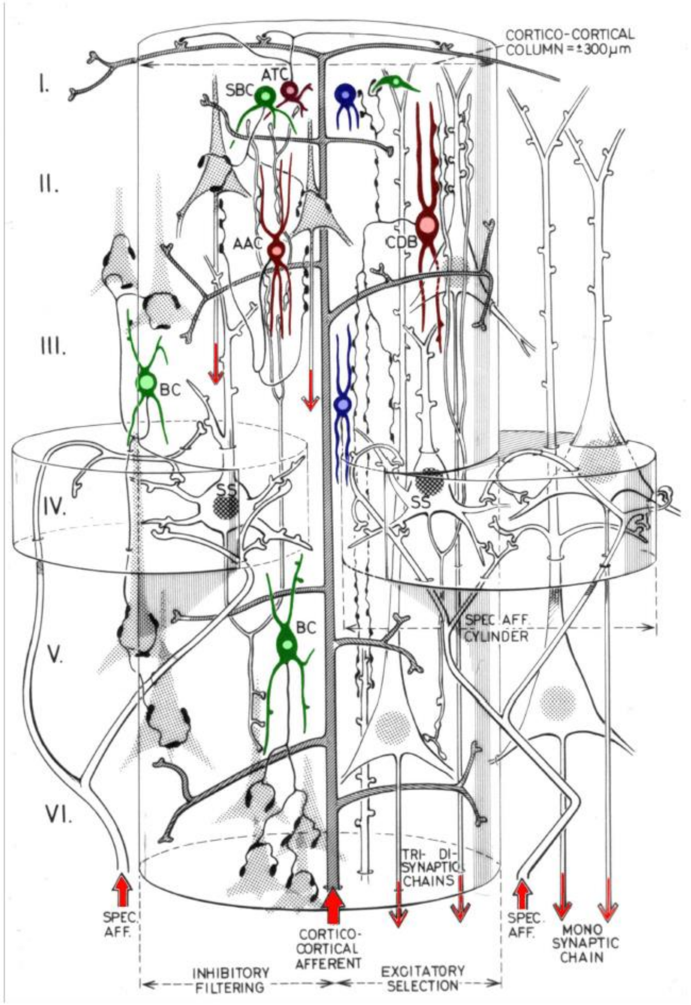
Many cell types (up to ~100 within a single brain region), variability within these classes (also at a functional level).

Common simplifications: only a few cell types, fixed or at most a few independently varied random parameters.

Complex networks of interactions







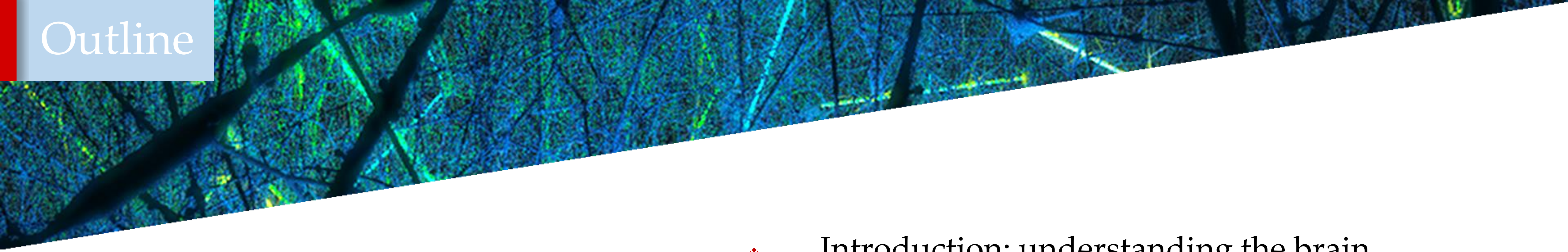
Not regular, but not random, at several different scales.

Common simplifications: homogeneous / random / fixed structured anatomical and functional connections, or calculated / simulated using plasticity rules that depend on neuronal activity.

Some fundamental problems in computational neuroscience

- How is information encoded by action potential trains? How to decode information encoded by action potential trains? Why does a specific part of the brain use a specific type of coding?
- How do neurons as information processing units function; specifically, what is the relation between the temporal and spatial pattern of the input and the spatial and temporal pattern of the output?
- How do neurons communicate, and what collective behaviors emerge in networks?
- How does cellular-level (synaptic) plasticity function? How can we understand behavioral-level learning? What is the connection between the two?

Outline



- Introduction: understanding the brain
- Systematic model construction and evaluation
    - Parameter optimization
    - Automated model validation
- Neural modeling example
    - Model of network dynamics, coding, and learning in the hippocampus

Motivation

- Our long-term goal: understand the dynamics and computations in hippocampal cells and circuits by combining experiments and models
- Detailed (data-driven / bottom-up) models can be very useful
- Problem: building high-quality data-driven models is difficult and slow
- Building on earlier efforts, we are developing and applying software tools for
  - parameter optimization
  - algorithm benchmarking
  - model validation
- These tools facilitate a more systematic and reproducible approach to modeling

Parameter optimization

- Detailed biophysical models typically contain several unknown parameters
- Older models were built using mostly ad hoc methods
- Automated parameter tuning has now become widespread
- Several different algorithms have been used
- Many other methods have been successfully applied in other domains
- Their performance (consistence, convergence speed etc.) has not been evaluated and compared systematically on neural problems
- We developed a neural optimization user interface (Neuroptimus) and a set of benchmarks to address these issues

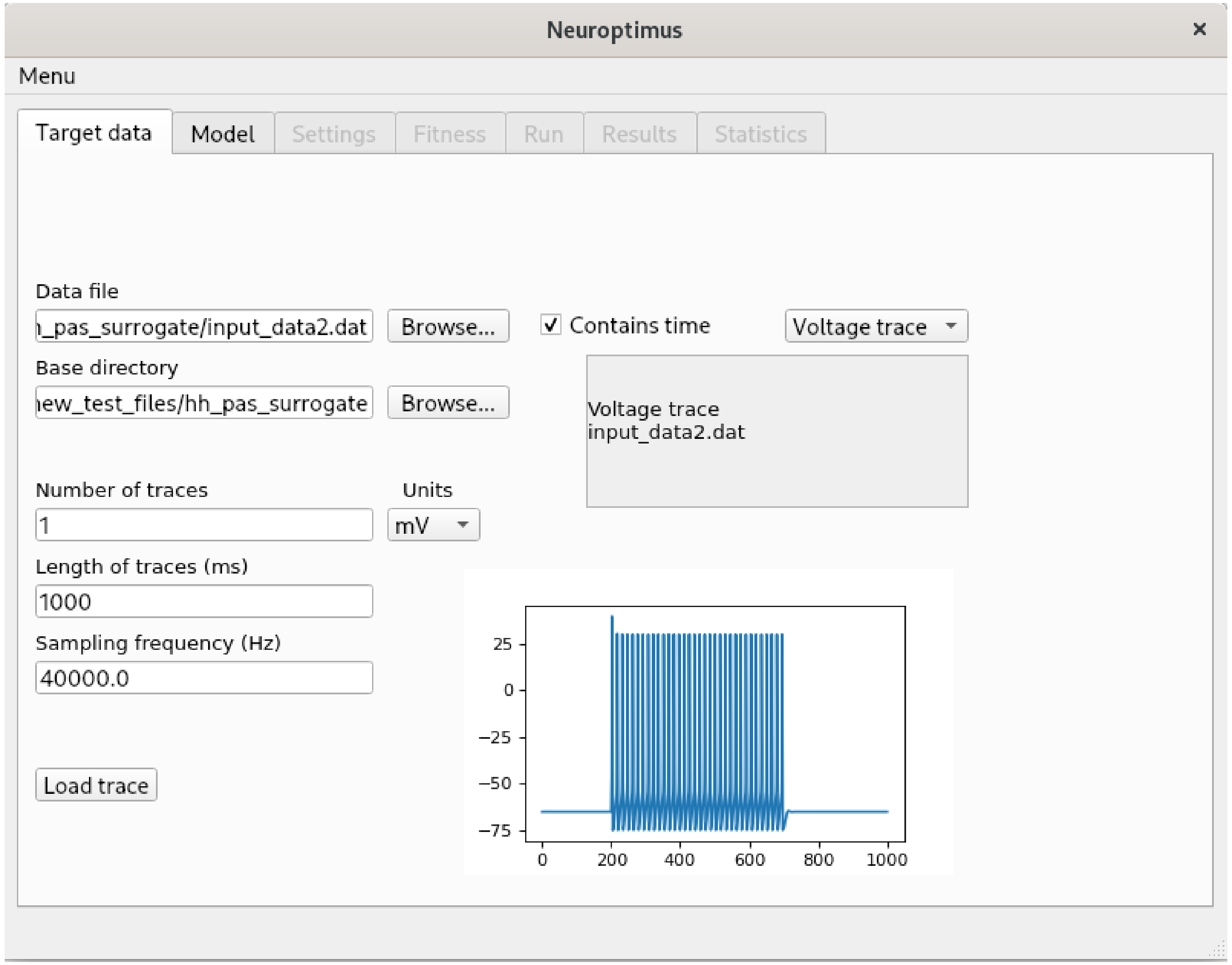
Neuroptimus: a tool for the automated fitting of model parameters

Based on our earlier tool Optimizer (https://github.com/KaliLab/optimizer)

Features:

- GUI supporting a variety of common scenarios
- Batch processing / command line operation
- Internal (Neuron) or external (black box) simulator
- Includes ~30 different algorithms from four different optimization modules (Inspyred, Pygmo, BluePyOpt, Scipy)
- Parallel execution enabled for many algorithms
- Supports current and voltage clamp, multiple traces, time series and abstract (feature) data
- Weighted combinations of various cost functions (eFEL + internal)
- Modular structure - makes it possible to add new error functions and optimization algorithms

The Neuroptimus GUI 1: Target data



Data file: _pas_surrogate/input_data2.dat

Base directory: ew_test_files/hh_pas_surrogate

✓ Contains time

Voltage trace

Voltage trace
input_data2.dat

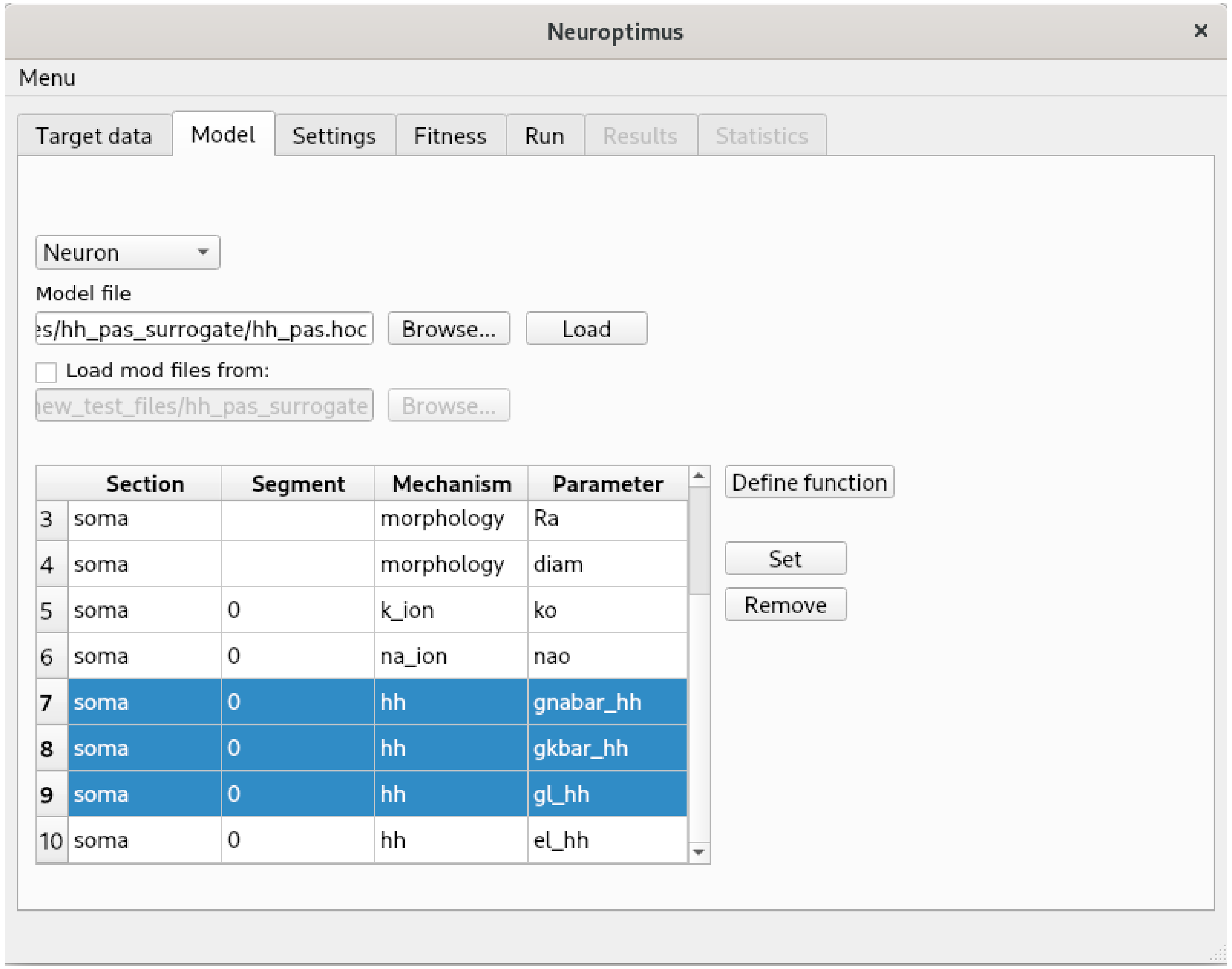
Number of traces: 1

Units: mV

Length of traces (ms): 1000

Sampling frequency (Hz): 40000.0

The Neuroptimus GUI 2: Model



Neuron

Model file

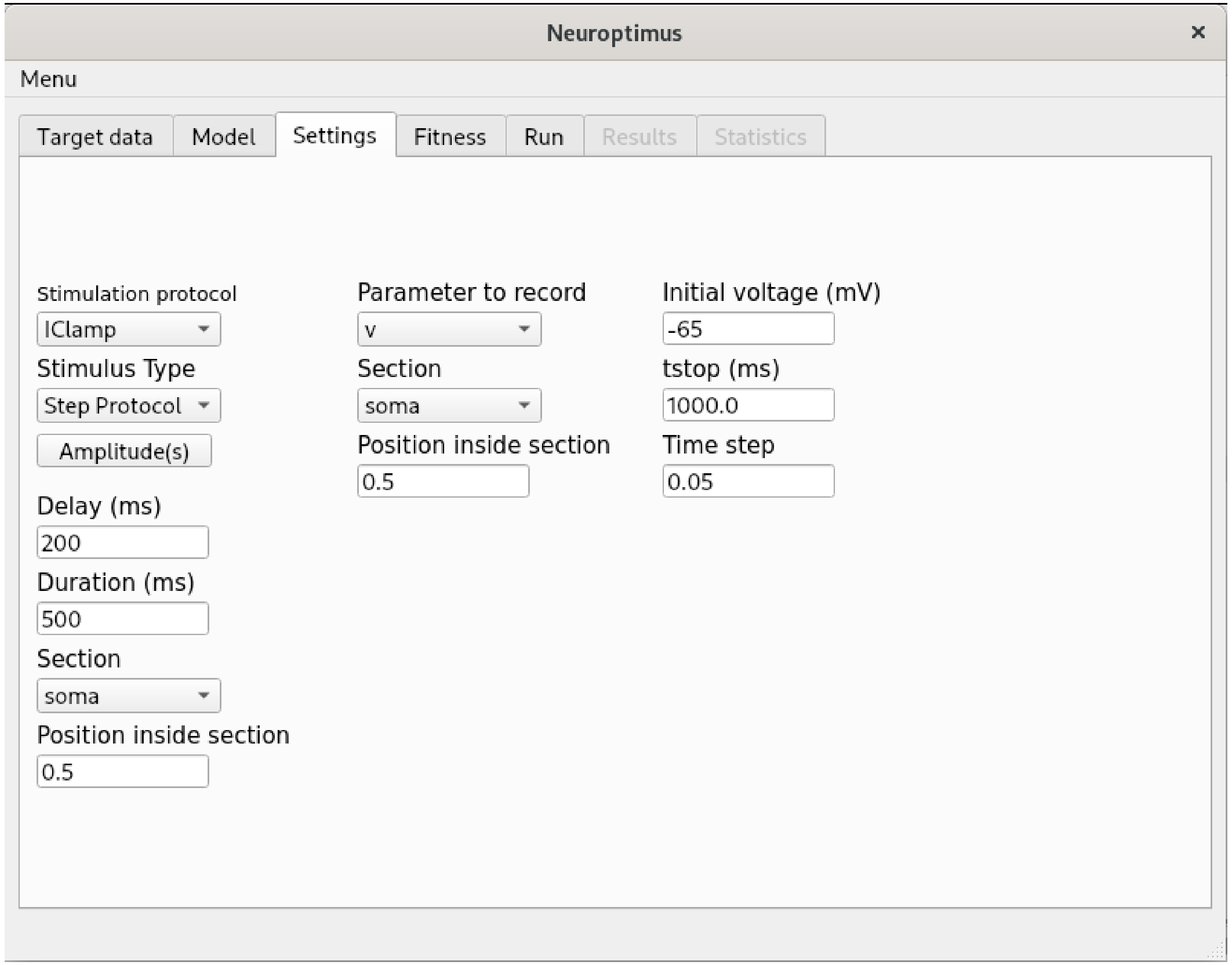
es/hh_pas_surrogate/hh_pas.hoc

☐ Load mod files from:

iew_test_files/hh_pas_surrogate

	Section	Segment	Mechanism	Parameter
3	soma		morphology	Ra
4	soma		morphology	diam
5	soma	0	k_ion	ko
6	soma	0	na_ion	nao
7	soma	0	hh	gnabar_hh
8	soma	0	hh	gkbar_hh
9	soma	0	hh	gl_hh
10	soma	0	hh	el_hh

The Neuroptimus GUI 3: Simulation settings

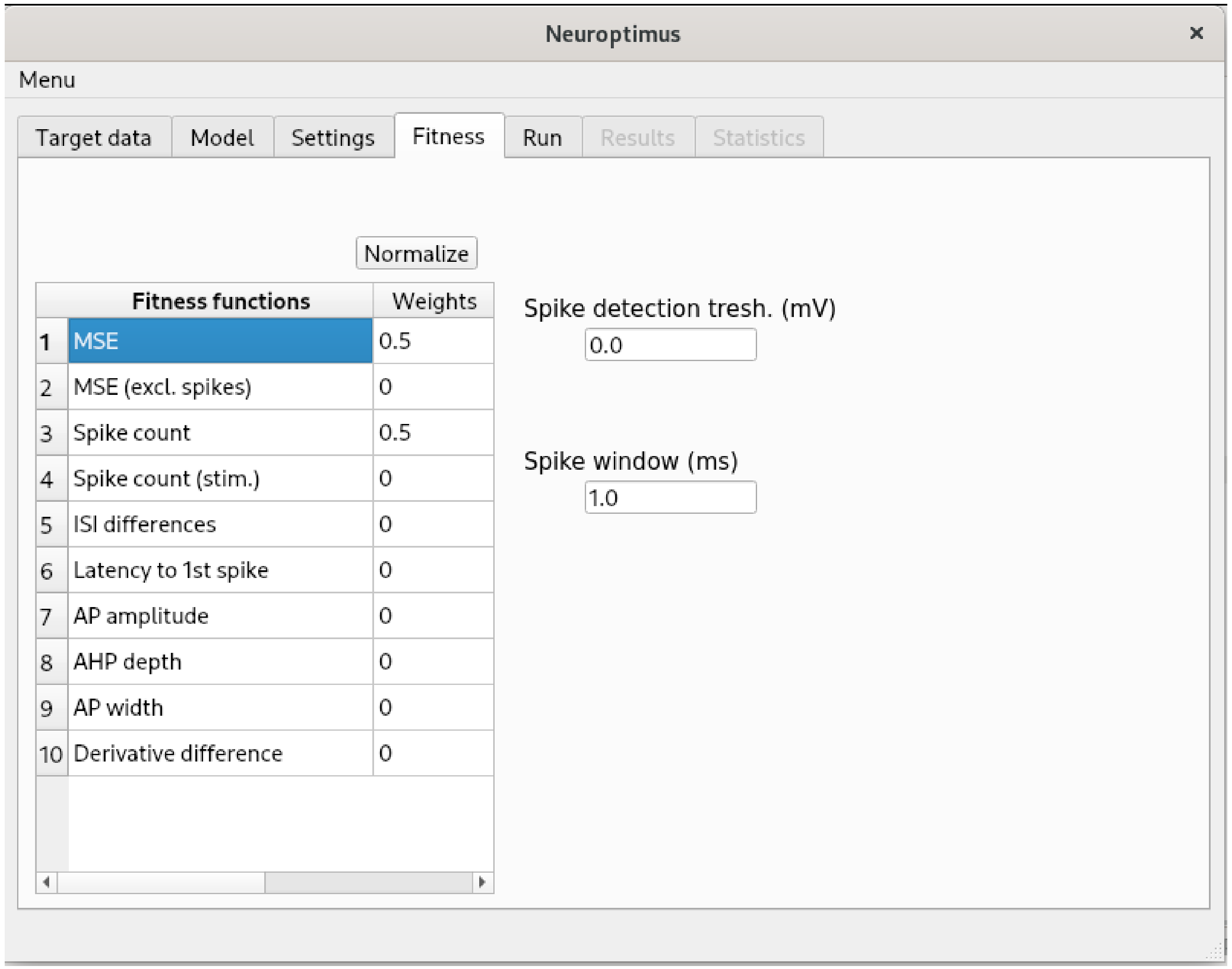


Stimulation protocol: IClamp
Stimulus Type: Step Protocol
Delay (ms): 200
Duration (ms): 500
Section: soma
Position inside section: 0.5

Parameter to record: v
Section: soma
Position inside section: 0.5

Initial voltage (mV): -65
tstop (ms): 1000.0
Time step: 0.05

The Neuroptimus GUI 4: Error functions

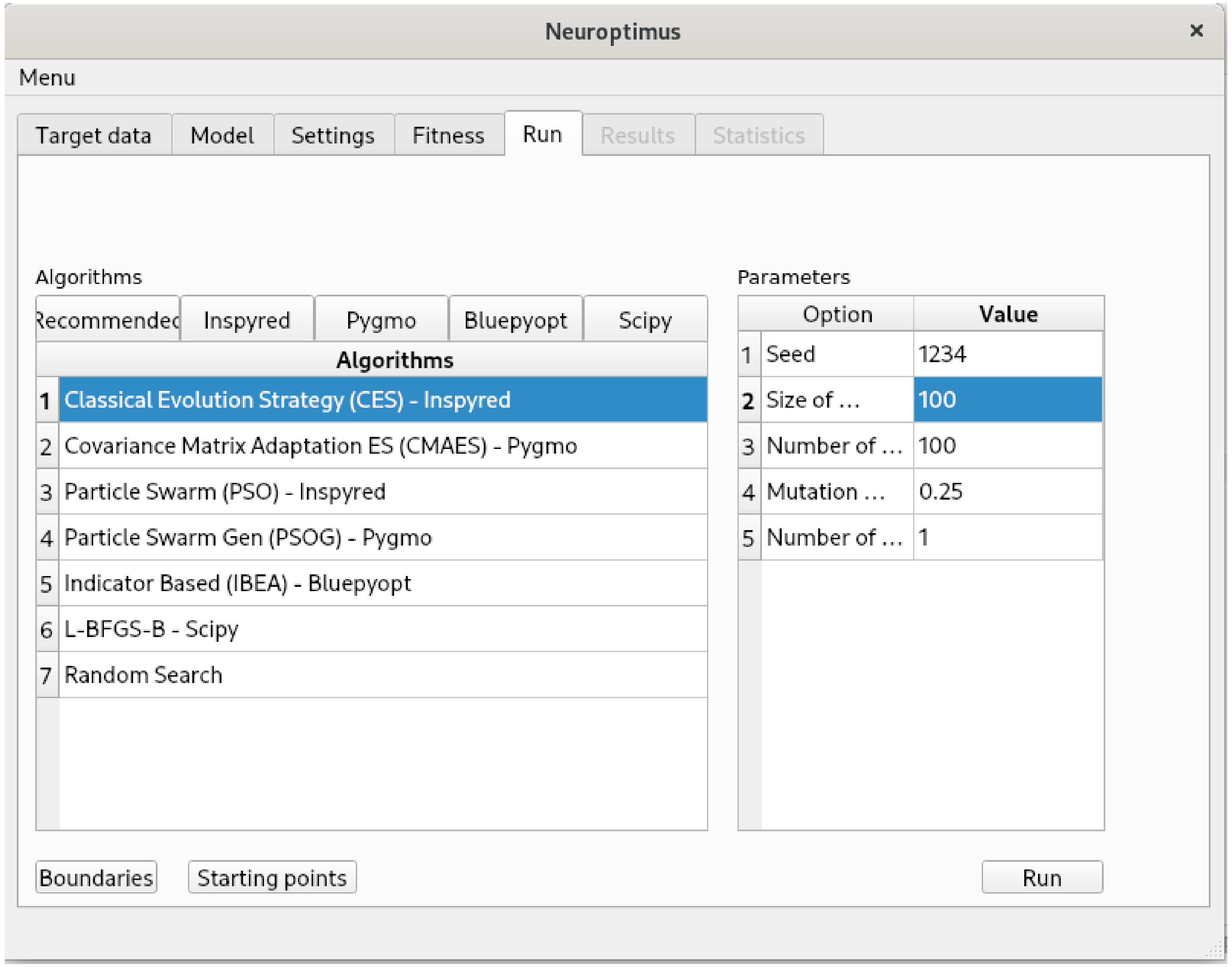


	Fitness functions	Weights
1	MSE	0.5
2	MSE (excl. spikes)	0
3	Spike count	0.5
4	Spike count (stim.)	0
5	ISI differences	0
6	Latency to 1st spike	0
7	AP amplitude	0
8	AHP depth	0
9	AP width	0
10	Derivative difference	0

Spike detection tresh. (mV)
0.0

Spike window (ms)
1.0

The Neuroptimus GUI 5: Algorithms



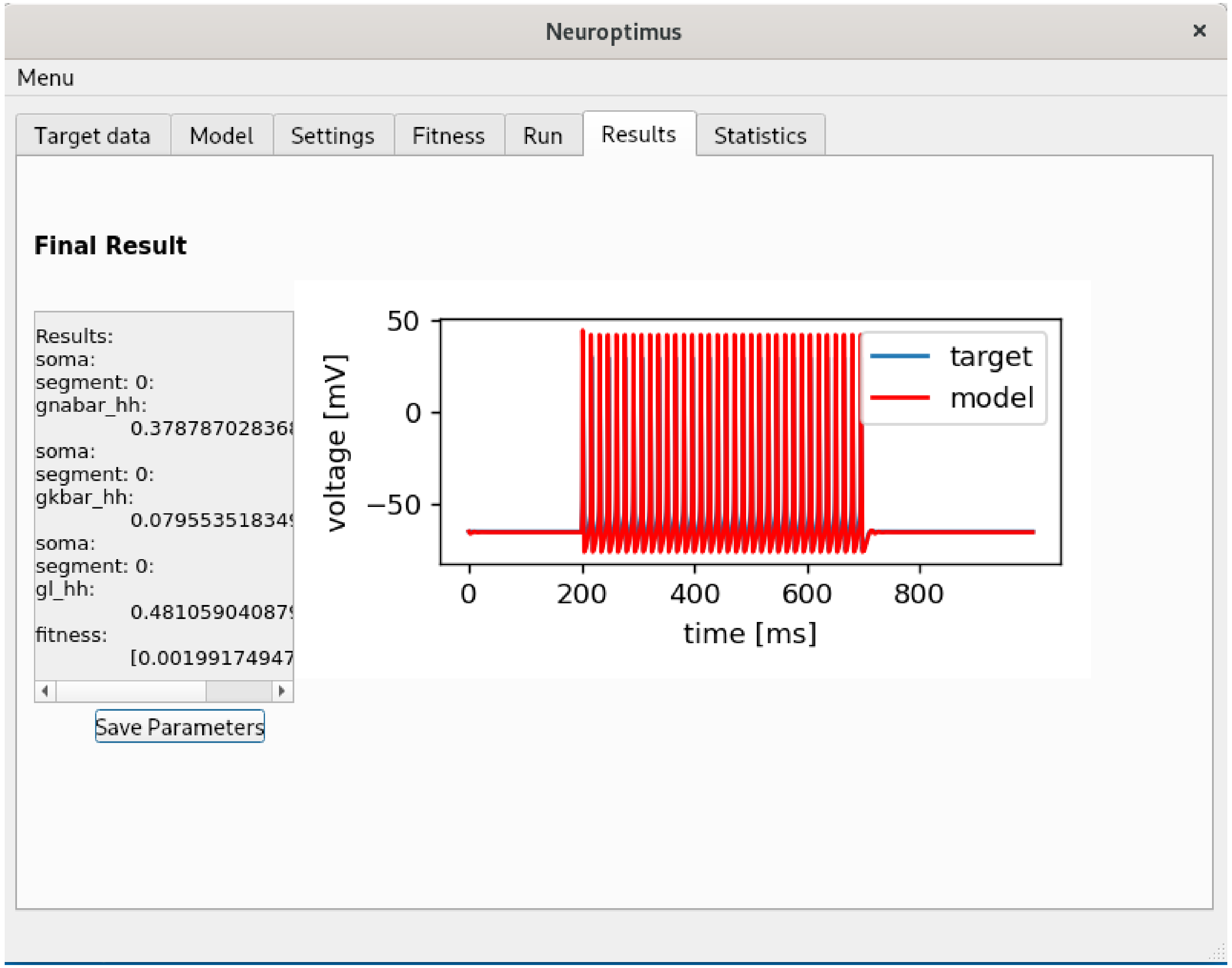
Algorithms

	Algorithms
1	Classical Evolution Strategy (CES) - Inspyred
2	Covariance Matrix Adaptation ES (CMAES) - Pygmo
3	Particle Swarm (PSO) - Inspyred
4	Particle Swarm Gen (PSOG) - Pygmo
5	Indicator Based (IBEA) - Bluepyopt
6	L-BFGS-B - Scipy
7	Random Search

Parameters

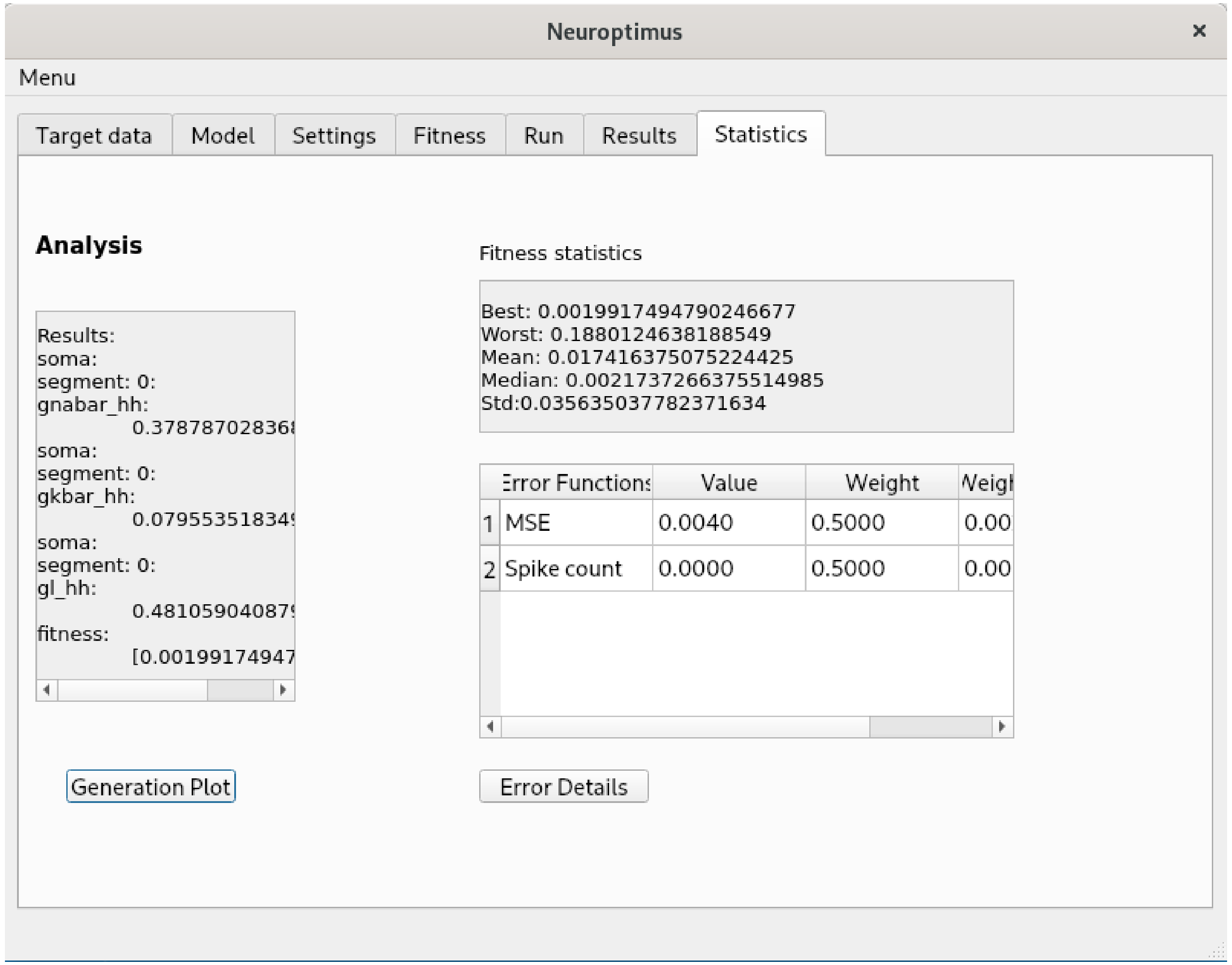
	Option	Value
1	Seed	1234
2	Size of …	100
3	Number of …	100
4	Mutation …	0.25
5	Number of …	1

The Neuroptimus GUI 6: Results



```
Results:
soma:
segment: 0:
gnabar_hh:
        0.37878702836
soma:
segment: 0:
gkbar_hh:
        0.07955351834
soma:
segment: 0:
gl_hh:
        0.48105904087
fitness:
        [0.00199174947
```

The Neuroptimus GUI 7: Statistics



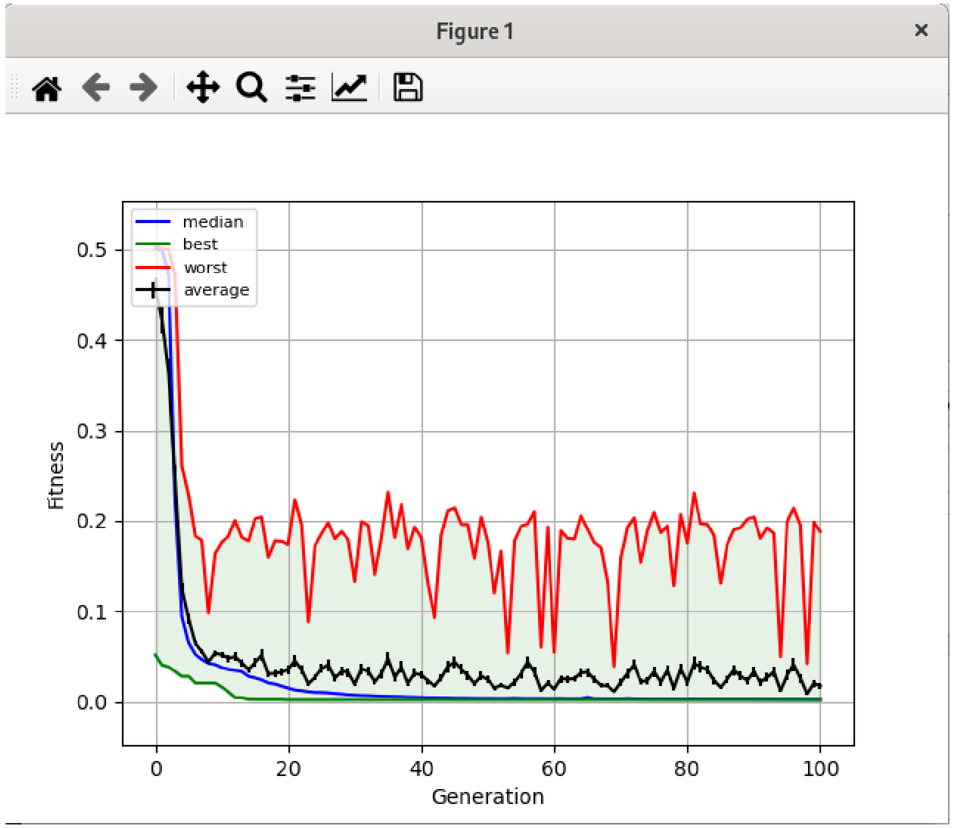
Analysis

Results:
soma:
segment: 0:
gnabar_hh:
0.37878702836
soma:
segment: 0:
gkbar_hh:
0.07955351834
soma:
segment: 0:
gl_hh:
0.48105904087
fitness:
[0.00199174947

Fitness statistics

```
Best: 0.0019917494790246677
Worst: 0.1880124638188549
Mean: 0.017416375075224425
Median: 0.0021737266375514985
Std:0.035635037782371634
```

	Error Functions	Value	Weight	Weigh
1	MSE	0.0040	0.5000	0.00
2	Spike count	0.0000	0.5000	0.00



Benchmarking optimization algorithms using Neuroptimus

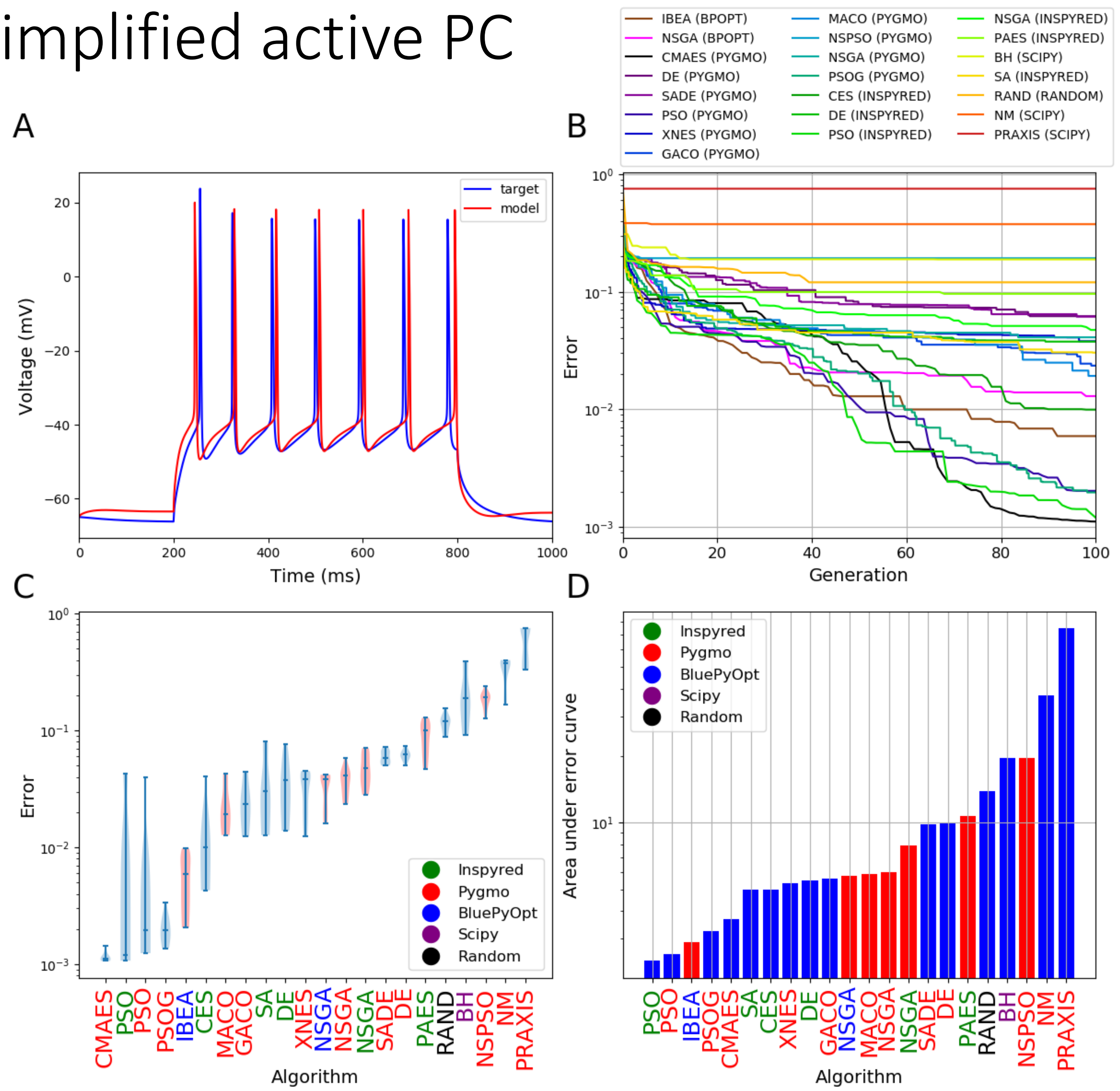
- Use cases for benchmarking are set up using the Neuroptimus GUI
- Each use case is then run multiple (10) times for every algorithm with different random seeds using the batch mode of Neuroptimus
- The evolution of the total error is extracted and evaluated automatically from each run
- Statistics are computed across runs
- Algorithms are compared according to
    - final error after 10,000 model evaluations
    - convergence speed (area under the error curve)
- Algorithms tested include
    - local and global methods
    - single-objective and multi-objective algorithms
    - gradient-based, random, evolutionary, swarm algorithms and other metaheuristics

Model fitting benchmark 4: Simplified active PC

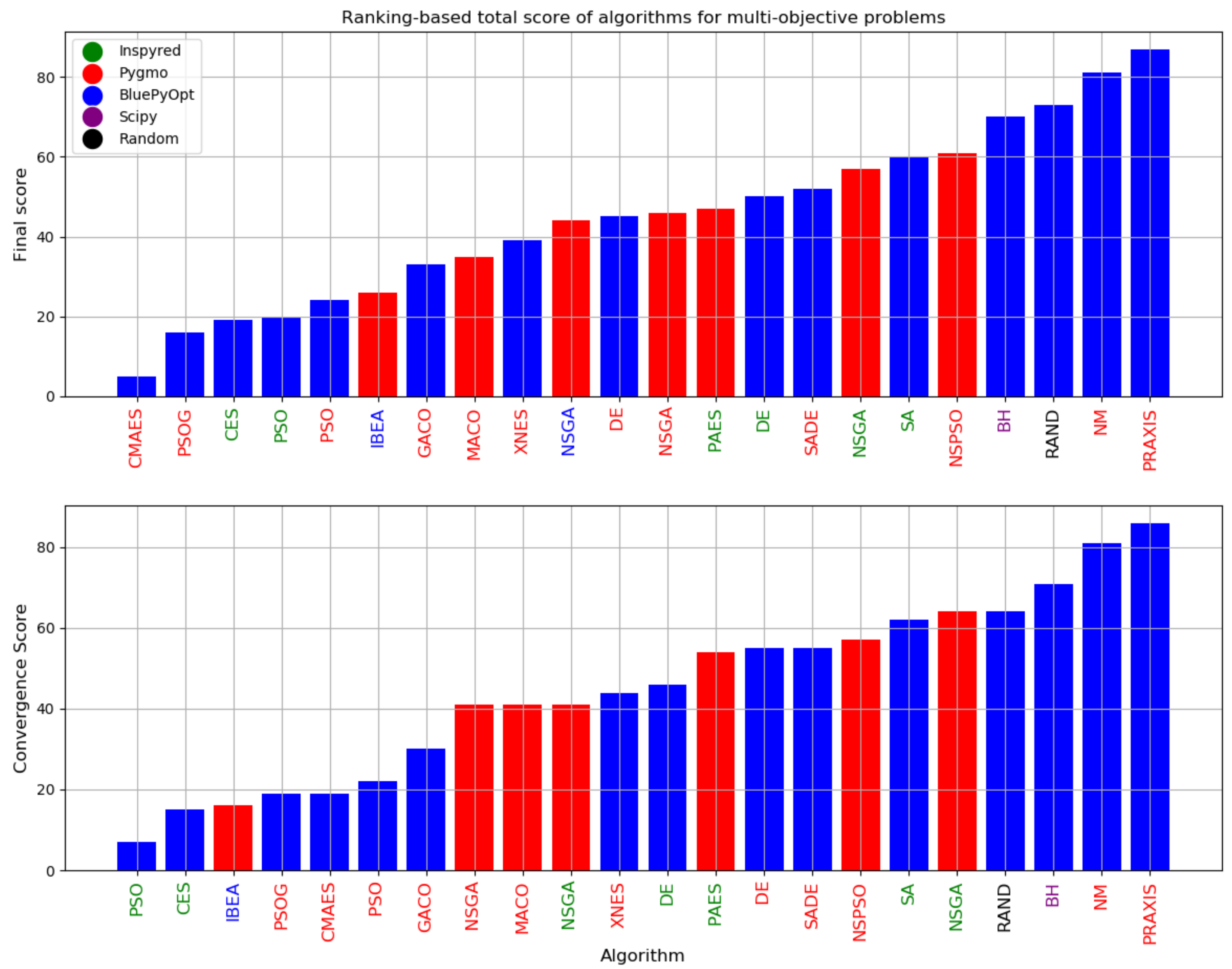
Fitting the somatic conductance densities of a simplified (6-compartmental) model to the voltage response of a morphologically and biophysically detailed CA1 pyramidal cell model to a current step stimulus.

(9 parameters)

(A) Comparison of the original (simulated) data (blue) and an example of a best-fitting trace (red). (B) Evolution of the median error value across generations in 22 different optimization algorithms. (C) Distribution of the final error scores across 10 independent repetitions for each algorithm. (D) Comparison of convergence speed based on the area under the median error curve. Red bars represent multi-objective algorithms.



Model fitting benchmarks: Final ranking



Parameter optimization: Conclusions

- Our software tool Neuroptimus supports a variety of neural model optimization use cases
    - Intuitive GUI for many basic tasks
    - Additional capabilities for power users
- Benchmarking many algorithms in four packages across six different use cases leads to some general recommendations:
    - Local algorithms (e.g., PRAXIS) are not suitable for more complex tasks
    - Generally well-performing algorithms include evolutionary strategies (CES, CMA-ES), particle swarm, and some multi-objective methods (IBEA)

Parameter optimization: Extensions

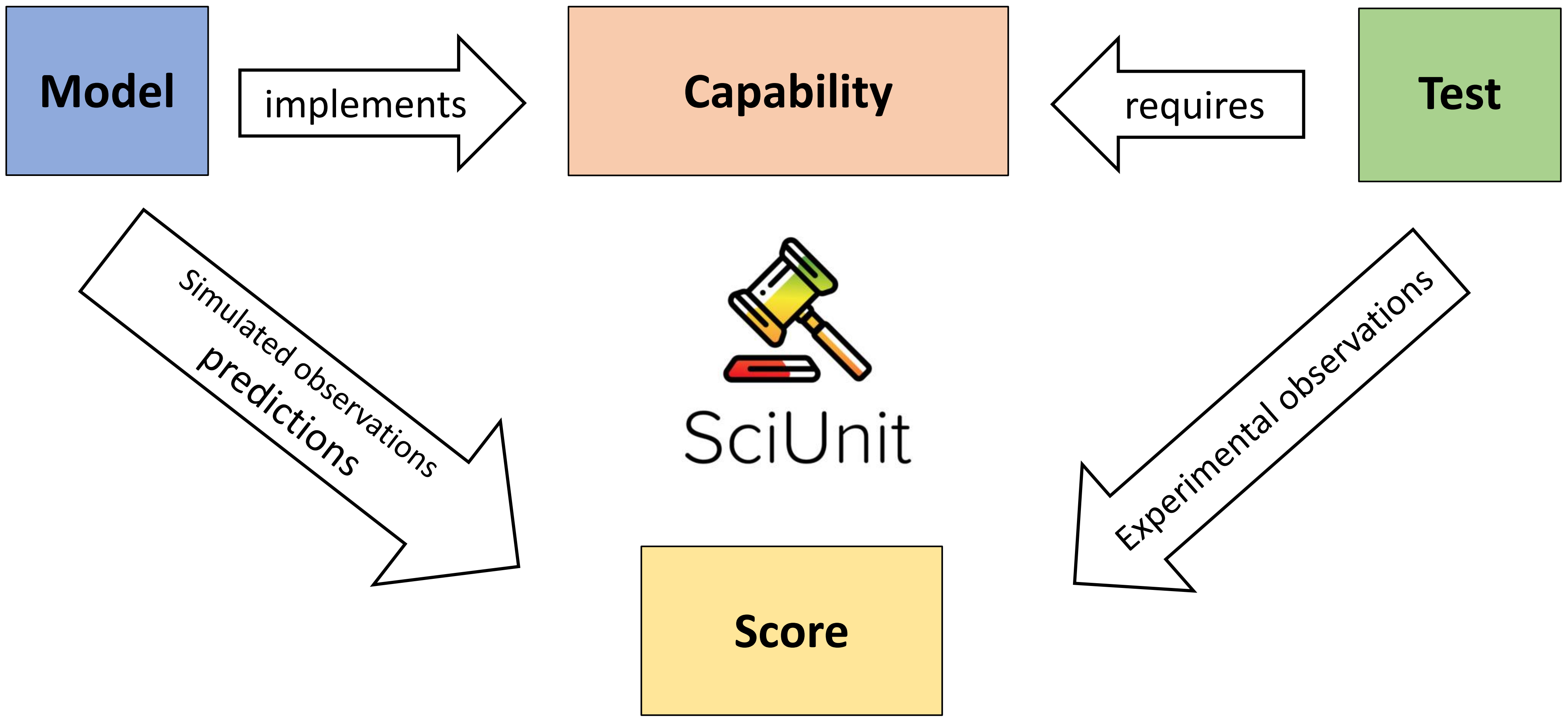
The basic fitting approach can be improved and extended to meet more specific requirements:

- For the reliable estimation of parameters of individual neurons, fitting can be replaced by proper (Bayesian) statistical inference
- To produce populations of neuronal models for network simulations, the variability of neurons needs to be modeled explicitly
  - What (multivariate) distribution of model parameters will lead to the appropriate (multivariate) distribution of physiological features?

Model validation - Motivation

- A large amount of experimental data are available for many cell types
- Many different models of the same cell type (136 CA1 PC models in ModelDB)
- Most models have been custom-built for a particular purpose, use a very limited set of constraints, and it is unknown how they behave outside their original context
- Model re-use is limited, and often leads to (undiscovered) “regression”
- Testing the behavior of models manually is difficult and error-prone
- We need automated validation suites!

Model validation framework: SciUnit



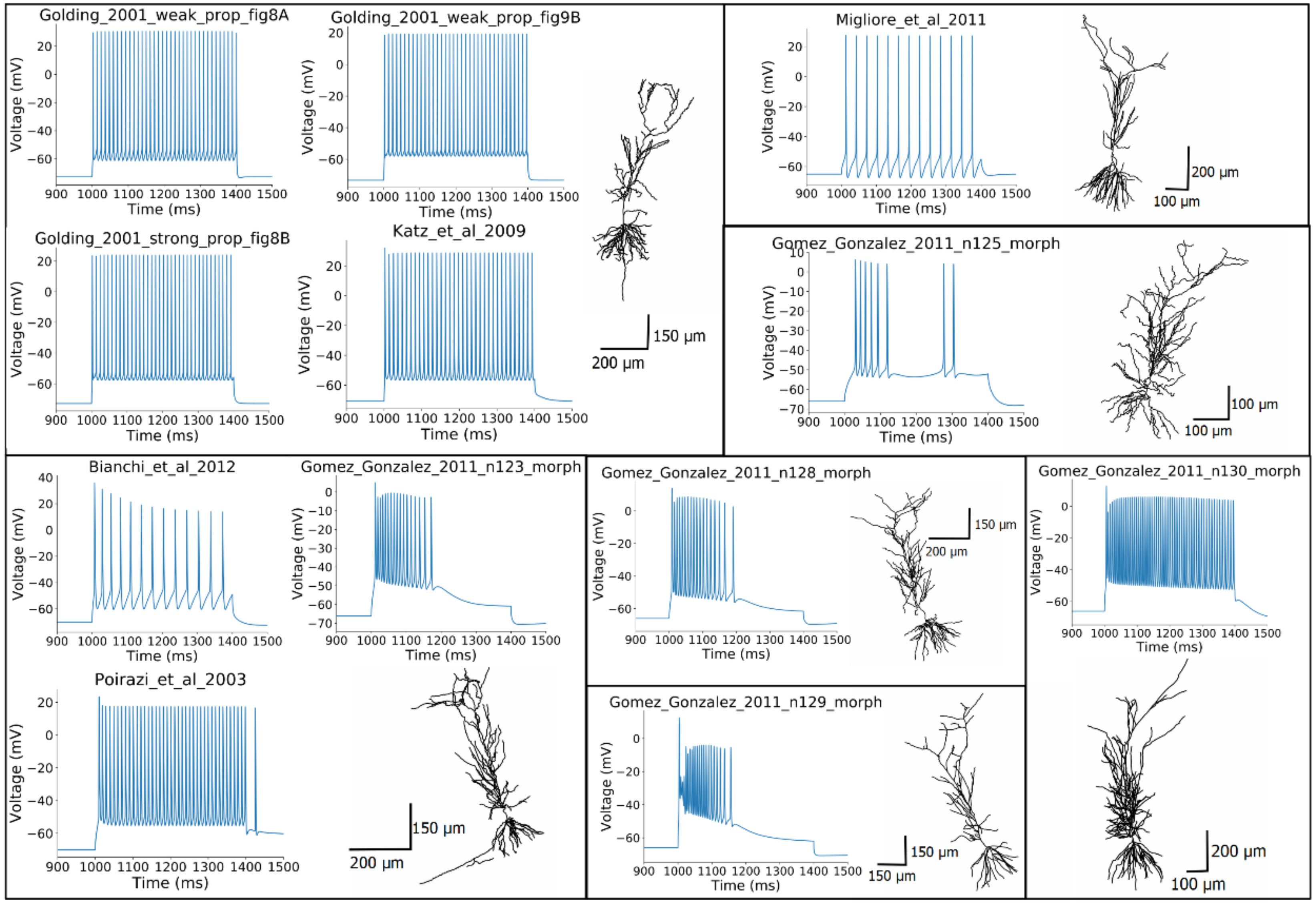
HippoUnit: Aims and design

- Electrophysiological validations for detailed hippocampal CA1 pyramidal cell models
- Simulations that mimic reported experimental protocols
- Tests implemented in Python, can be run from Jupyter Notebooks
- Feature extraction:
  - Own functions
  - Electrophys Feature Extraction Library (eFEL) of the BBP (Van Geit et al. 2016)
- Quantitative evaluation
  - observation (experimental data) ←→ prediction (model behavior)
  - Feature-based error function (Druckmann et al. 2007).
- Output:
  - Feature and score (error) values extracted from the models
  - Final score that is the sum of the errors of all the features tested by the given test
  - figures which illustrate the model's behavior and the extracted feature values

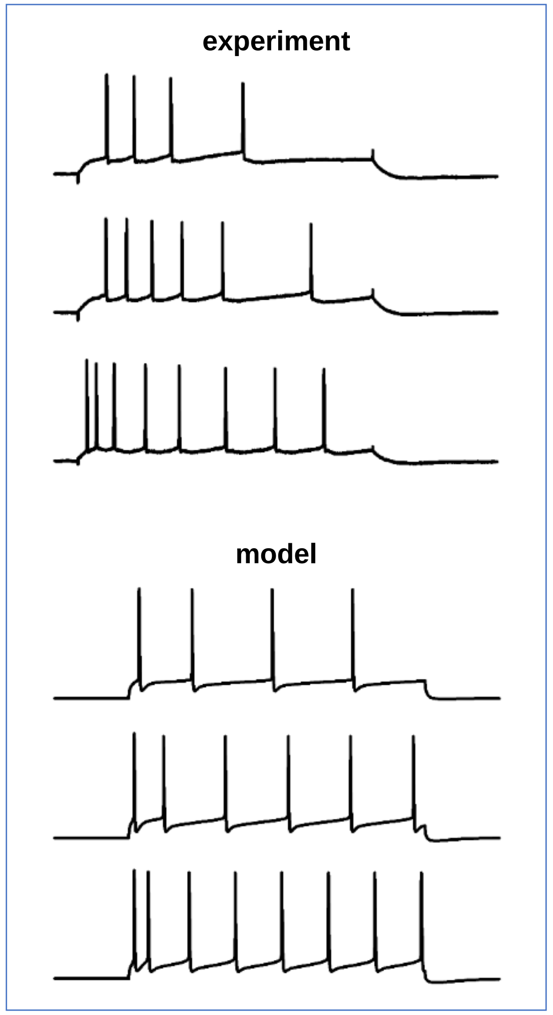
Tests of HippoUnit

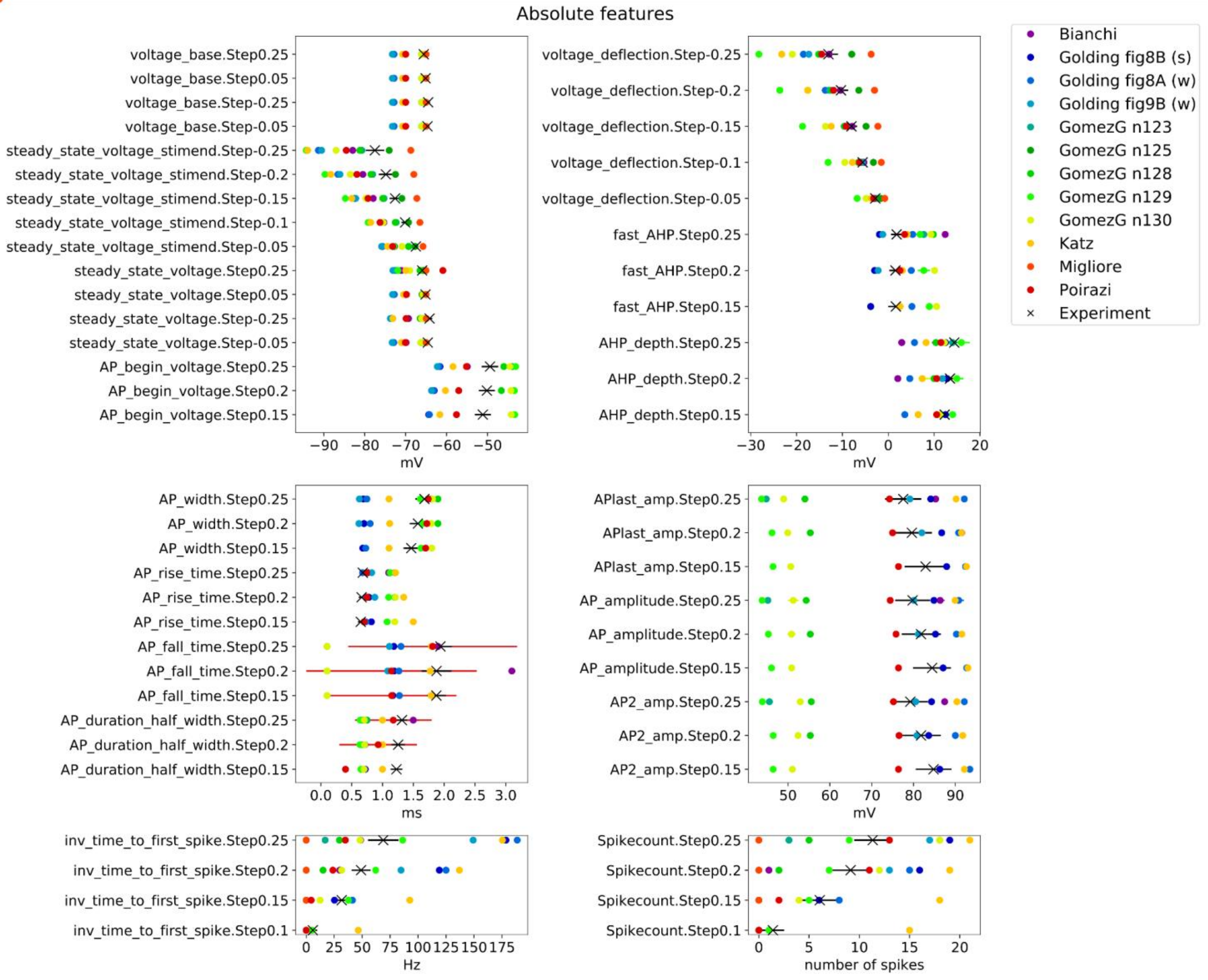
- The **Somatic Features Test** evaluates (using eFEL) and compares to experimental data the features of the somatic membrane potential response to somatic current injections of varying amplitudes.
- The **PSP Attenuation Test** evaluates how much the post synaptic potential attenuates from the main apical dendrite to the soma. (experimental data from Magee & Cook 2000)
- The **Back-propagating AP Test** Evaluates the mode and efficacy of back-propagating action potentials at different locations on the apical trunk. (experimental data from Golding et al. 2001)
- The **Depolarization Block Test** aims to determine whether the model enters depolarization block in response to prolonged, high intensity somatic current stimuli. (experimental data from Bianchi et al. 2012)
- The **Oblique Integration Test** probes the integration properties of the radial oblique dendrites for increasing number of synchronous and asynchronous inputs. (experimental data from Losonczy, Magee 2006)

CA1 pyramidal cells from the literature



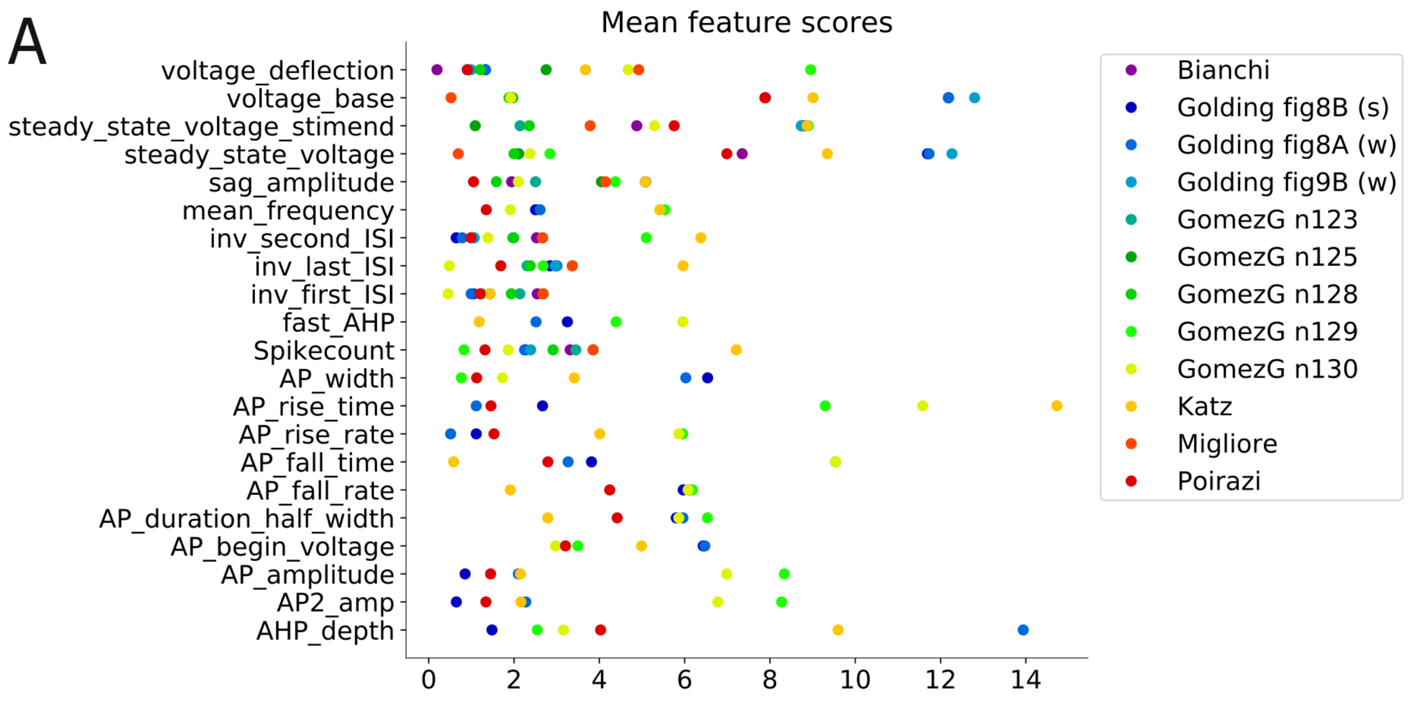
The **Somatic Features Test** evaluates (using eFEL) and compares to experimental data the features of the somatic membrane potential response to somatic current injections of varying amplitudes.



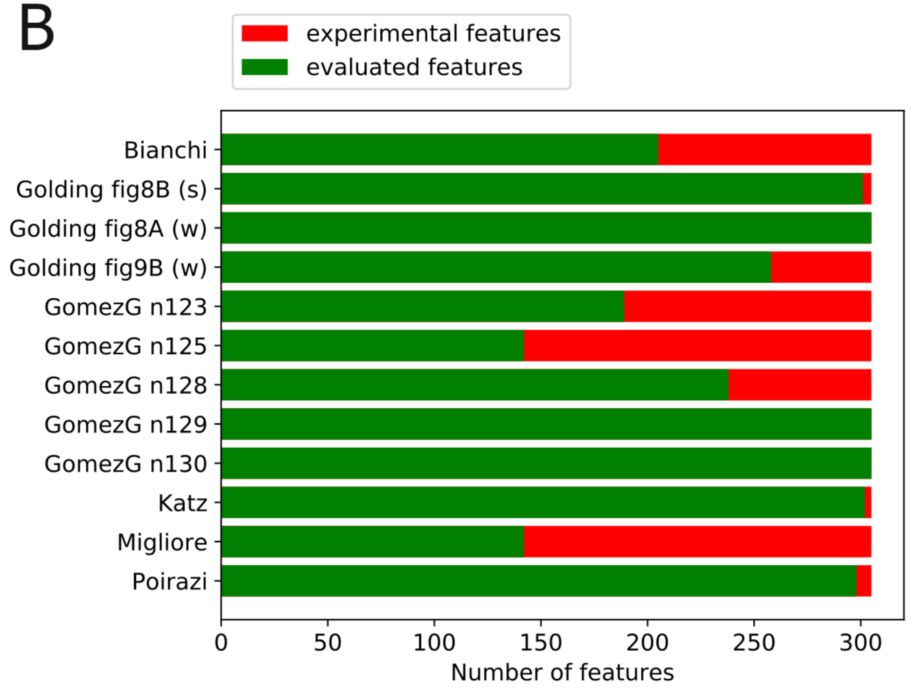


The **Somatic Features Test** evaluates (using eFEL) and compares to experimental data the features of the somatic membrane potential response to somatic current injections of varying amplitudes.

A

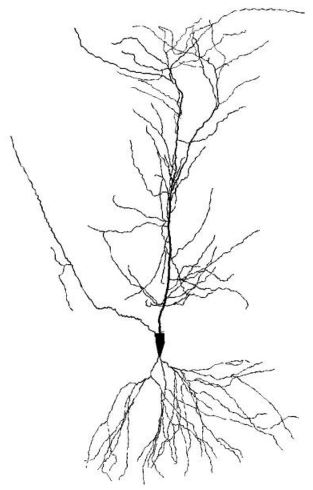


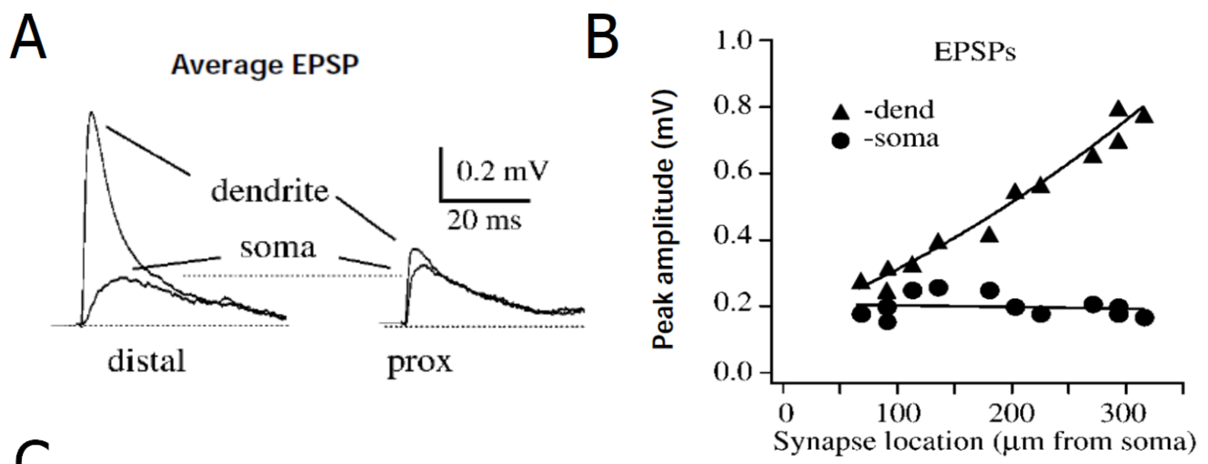
B

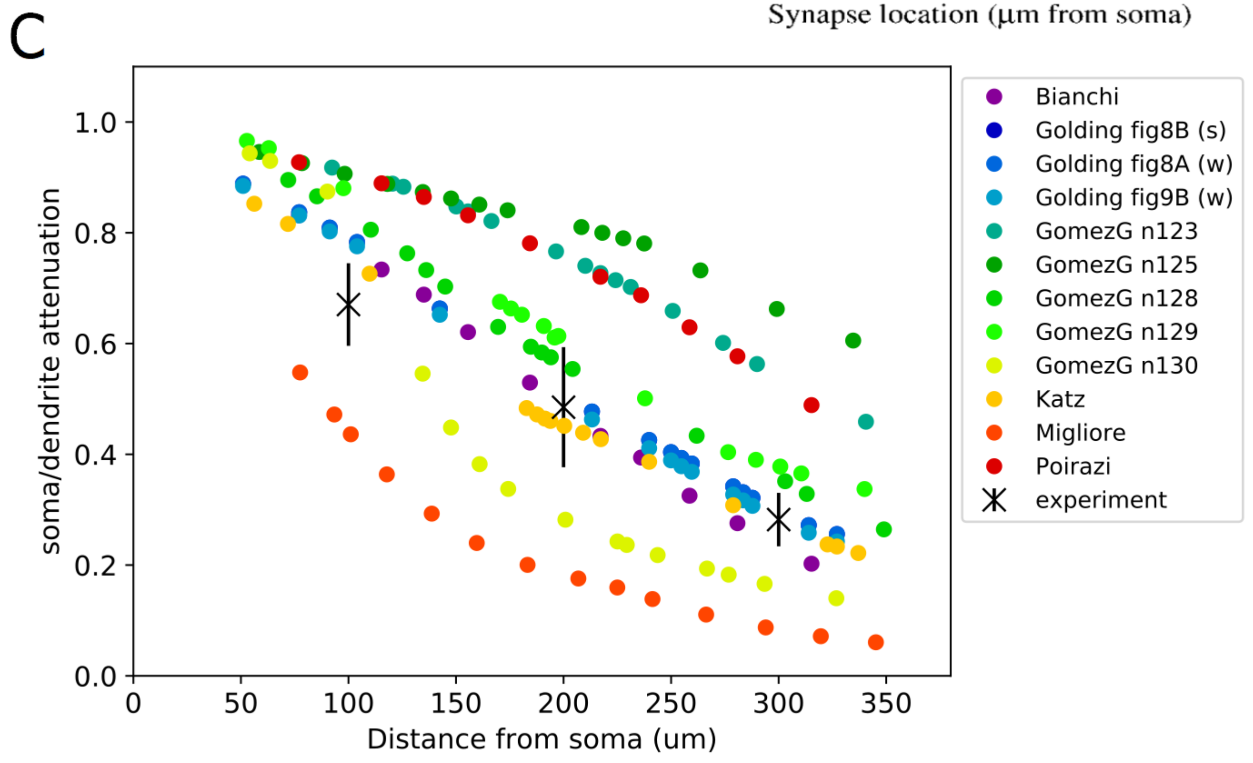


(A) Mean feature scores (# sd) of the different models. Feature score values are averaged over the different input step amplitudes.

(B) Number of experimental features attempted to be evaluated for the models (red), and number of successfully evaluated features (green).



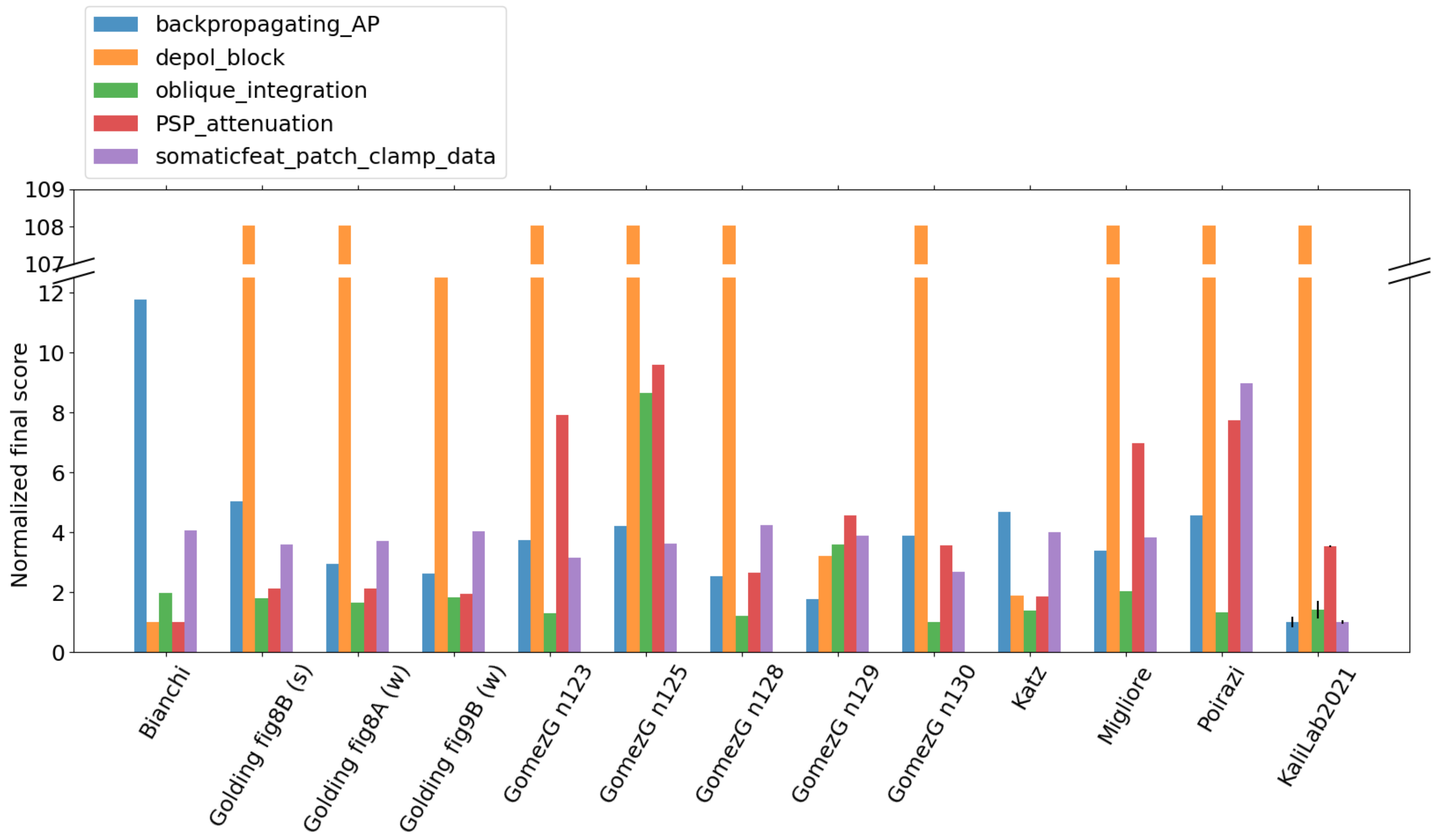




The **PSP Attenuation Test** evaluates how much the post synaptic potential (using a synaptic current stimulus) attenuates from the main apical dendrite (different distances) to the soma.

(data from Magee & Cook 2000)

Overall characterization and model comparison based on all tests of HippoUnit



Normalized final scores achieved by the different published models on the various tests of HippoUnit. The final scores of each test are normalized by dividing the scores of each model by the best achieved score on the given test.

The last model on the right is the current version of a new model developed in our lab using a systematic approach – automated optimization with Neuroptimus, and validation with HippoUnit.

Applications of HippoUnit

- Evaluation and comparison of existing models
  - provides independent and standardized verification of the behavior of the models,
  - allows researchers to learn more about models published by other groups,
  - and to judge which existing models show a good match to the experimental data in the domains that they care about, and thus to decide whether they could re-use one of the existing models in their own research.
- During model development - allows researchers to easily evaluate models in relation to the relevant experimental data after every iteration of model adjustment
  - avoid “regressions”
- Could enable the creation of “community models” through the iterative refinement of models in an open collaboration of multiple research teams.

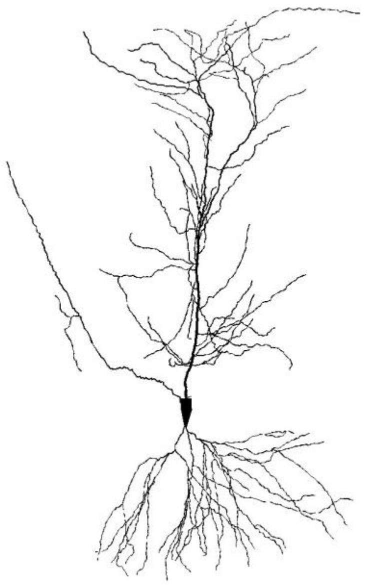
Validation – Interpretation and limitations

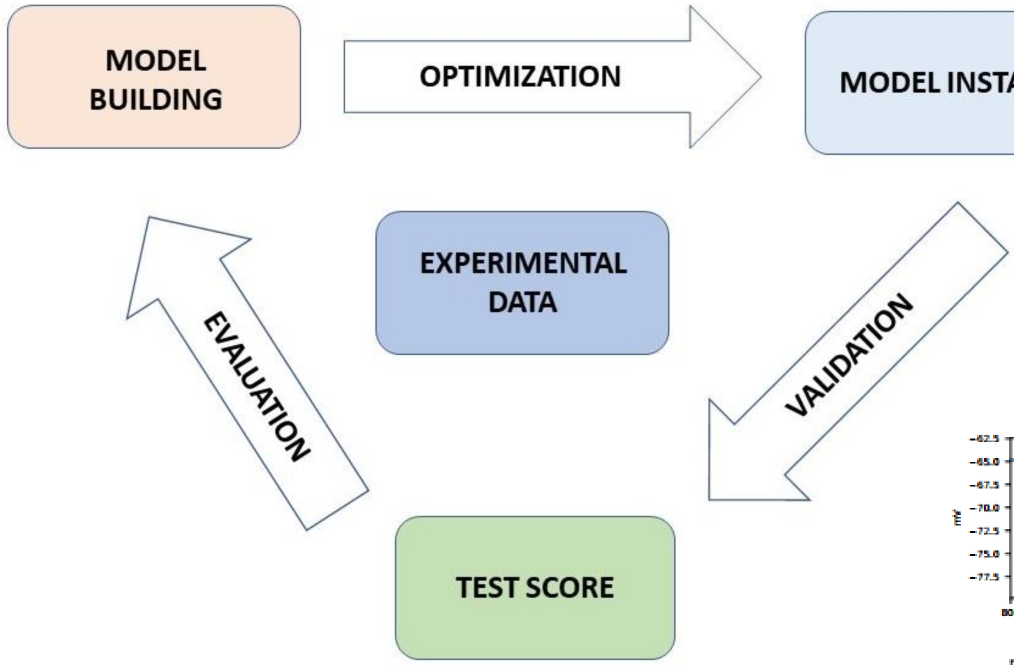
- A high error score on a particular test does not imply that the model is bad:
  - Perhaps the data are noisy or biased
  - Perhaps the test failed
  - Perhaps the model was designed to mimic different experimental conditions (species, strain, age, temperature, recording technique, etc.)
  - Real neurons are diverse (they are not like a single “average” neuron) – even large scores could match some real cells (especially with long-tailed distributions)
  - The model could be great for other purposes
- Validations should not be used blindly
  - Consider modeling goals (not all results may be relevant, or the test should be adapted)
  - Check outputs other than the final score
- Models often need to be adapted to run the tests – well-designed, standardized model formats could help

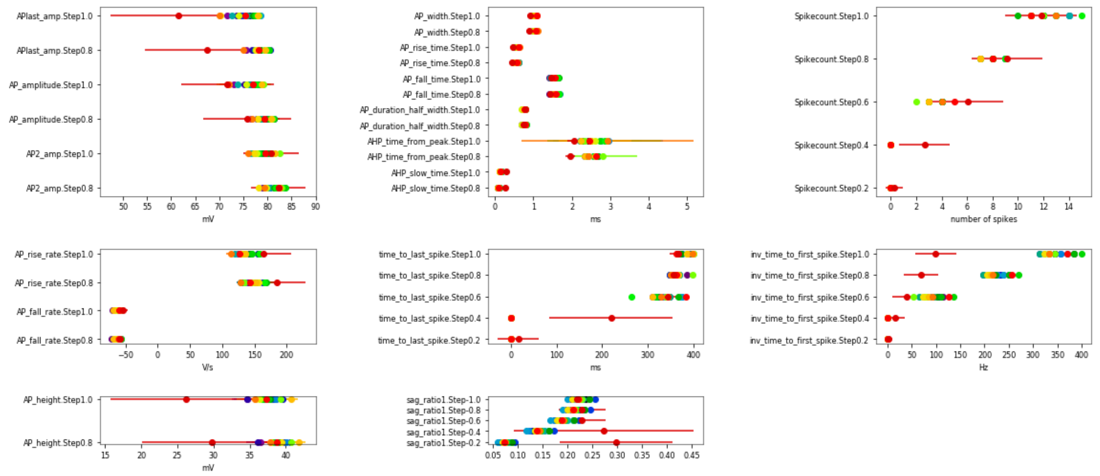
Multi-purpose models

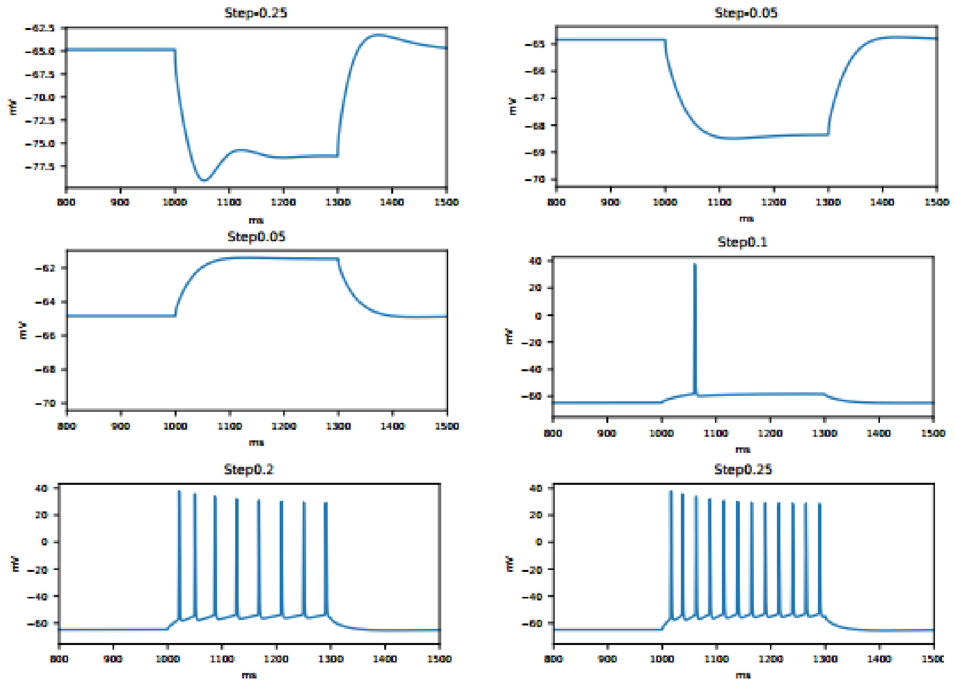
- Good models allow us to make new predictions
- Generalization:
  - Performance on untrained features
  - Response to new inputs from the same class
  - Performance in new paradigms
- Lessons from machine learning: how to build (train), validate, and select models
  - Validate (ideally, also train) in all domains where you want to make predictions (interpolation is much easier than extrapolation)
  - Do not use all the data for training – you also need data for validation (to avoid overfitting and select models that make good predictions)

Our modeling workflow

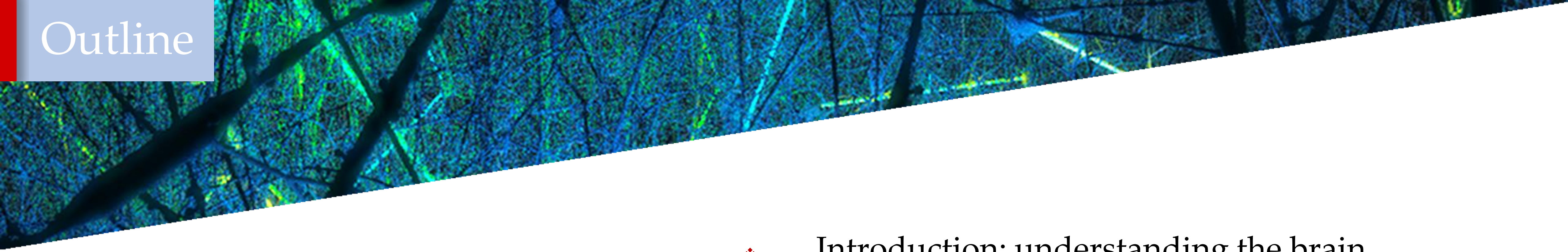






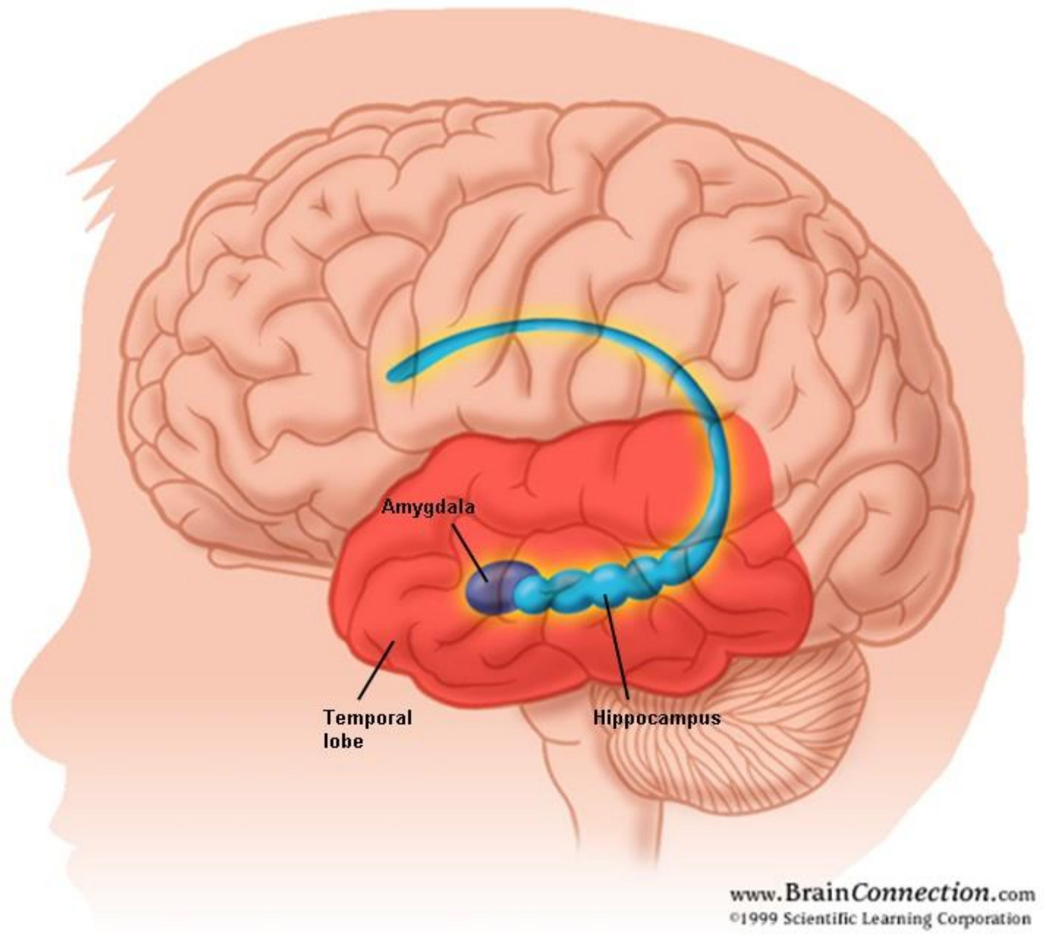


Outline



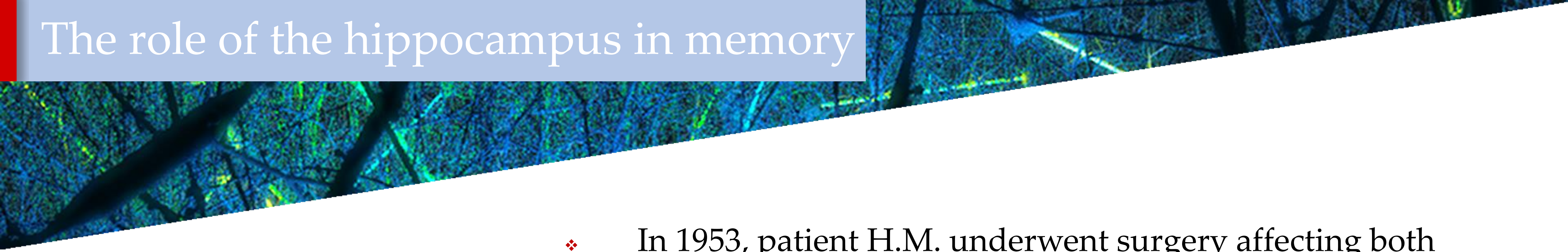
- Introduction: understanding the brain
- Systematic model construction and evaluation
    - Parameter optimization
    - Automated model validation
- Neural modeling example
    - Model of network dynamics, coding, and learning in the hippocampus

Functions of the hippocampus



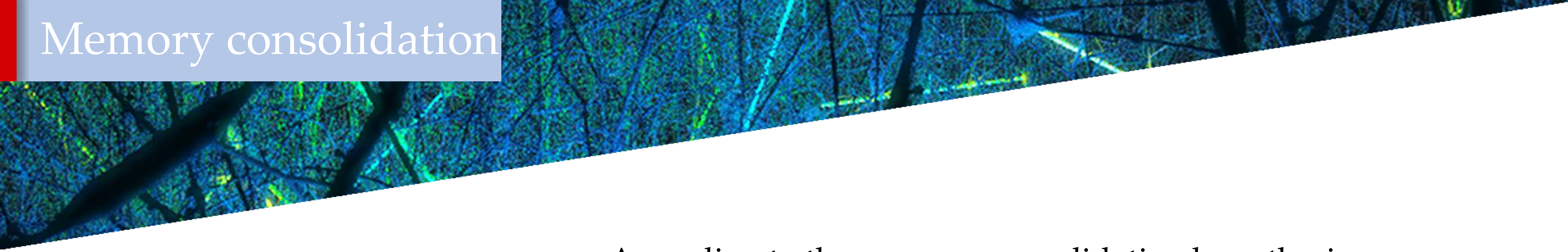
- Long-term "declarative" memory
- Spatial representation and navigation
- Formation of complex representations
- Others…

The role of the hippocampus in memory



- In 1953, patient H.M. underwent surgery affecting both hippocampi, and developed a very severe memory deficit.
- The hippocampus is critical for “declarative” (event, fact) memory, but not for “procedural” (e.g., perceptual and motor skill) memory.
- Amnesia has an “anterograde” component (affecting new learning) and a “retrograde” component (affecting earlier memories).
- Retrograde amnesia is often temporally graded (affects recent memories more than old ones).

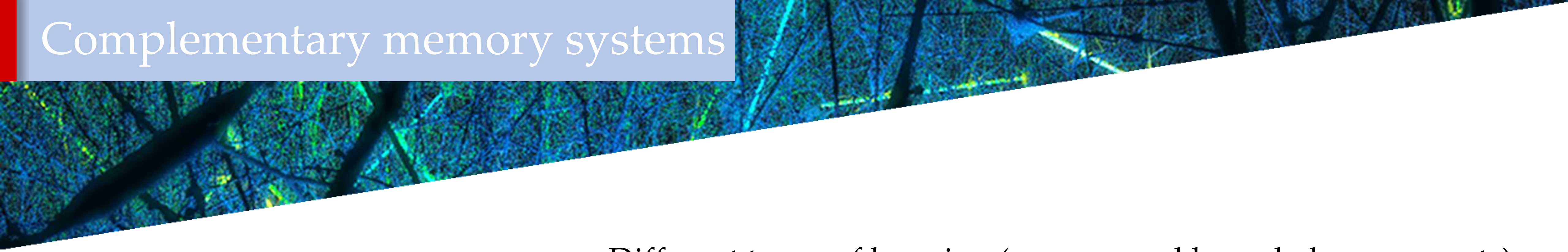
These observations led to the memory consolidation hypothesis.



Memory consolidation

According to the memory consolidation hypothesis

- the hippocampus stores information only temporarily
- memory traces are eventually “transferred” to neocortex
- the mechanism may involve the “off-line" reactivation of hippocampal memory traces



Complementary memory systems

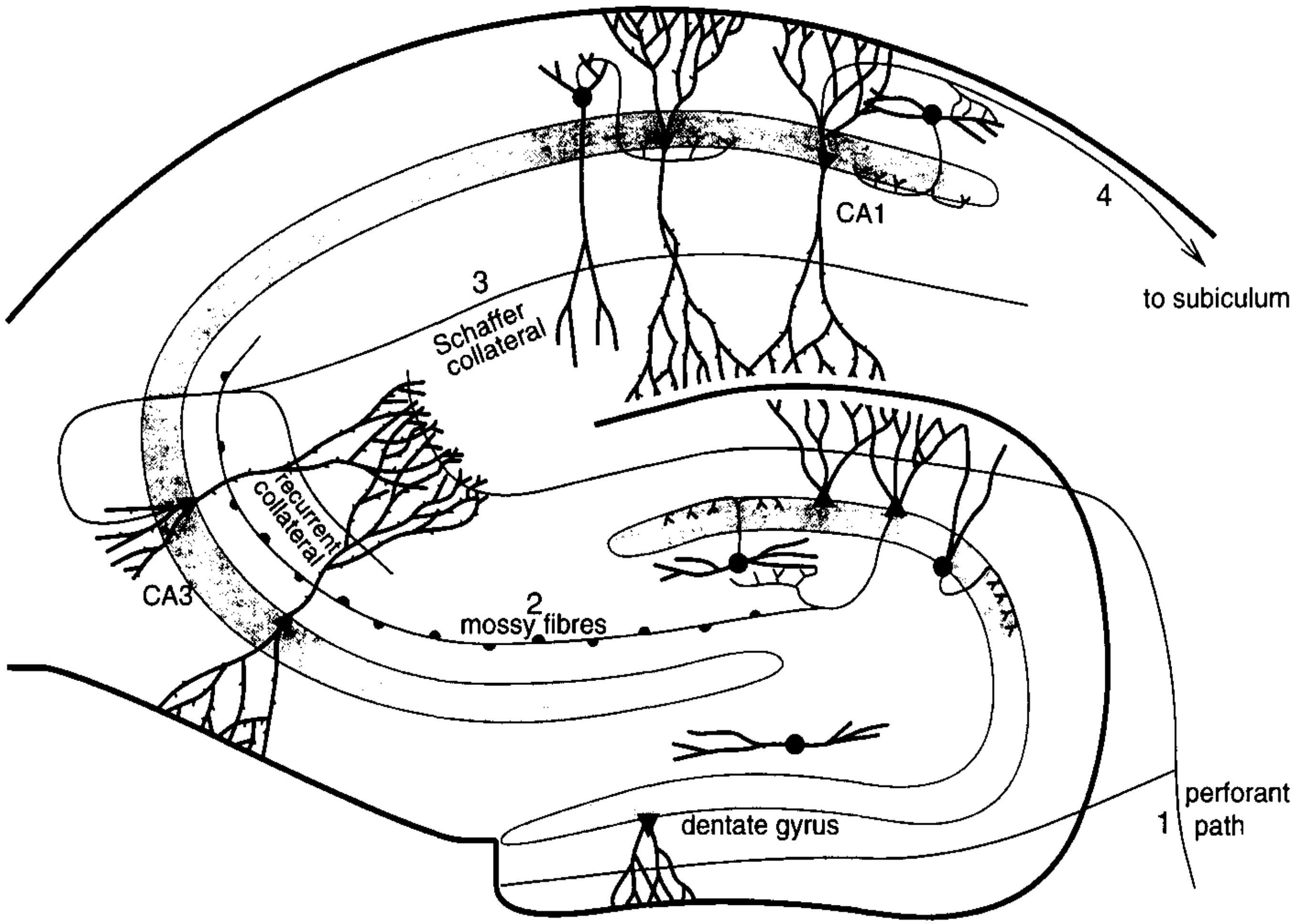
Different types of learning (e.g., general knowledge vs. events) may require different approaches.

According to the theory of complementary memory systems

- neocortex is specialized in slow, incremental (statistical) learning of general information
- the hippocampus is responsible for the rapid (“one-shot”) learning of arbitrary combinations of stimuli

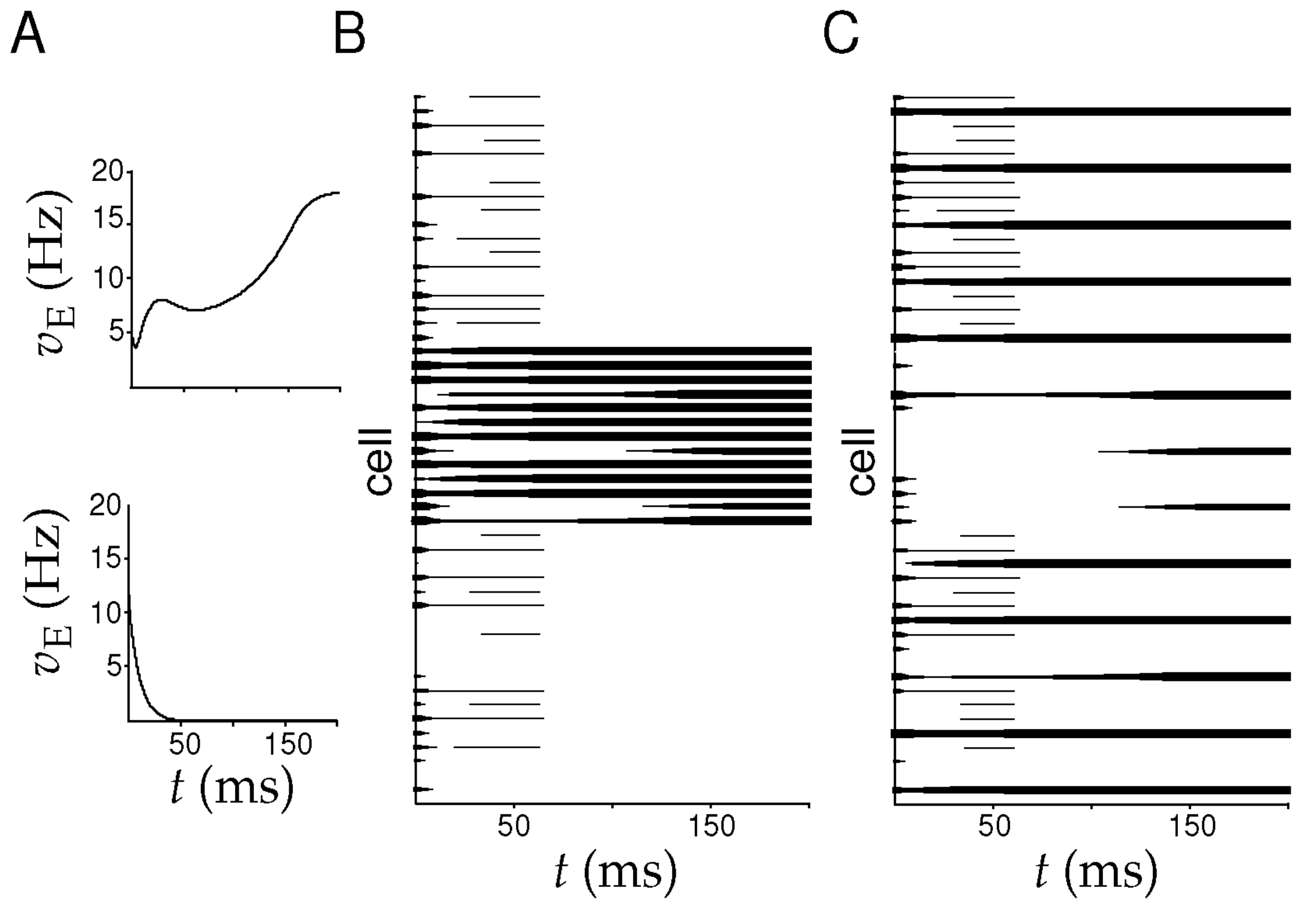
McClelland, McNaughton, and O’Reilly, 1995

The hippocampus as a long-term memory store



The recurrent collateral connections of area CA3 may implement an auto-associative memory network.

Associative (long-term) memory

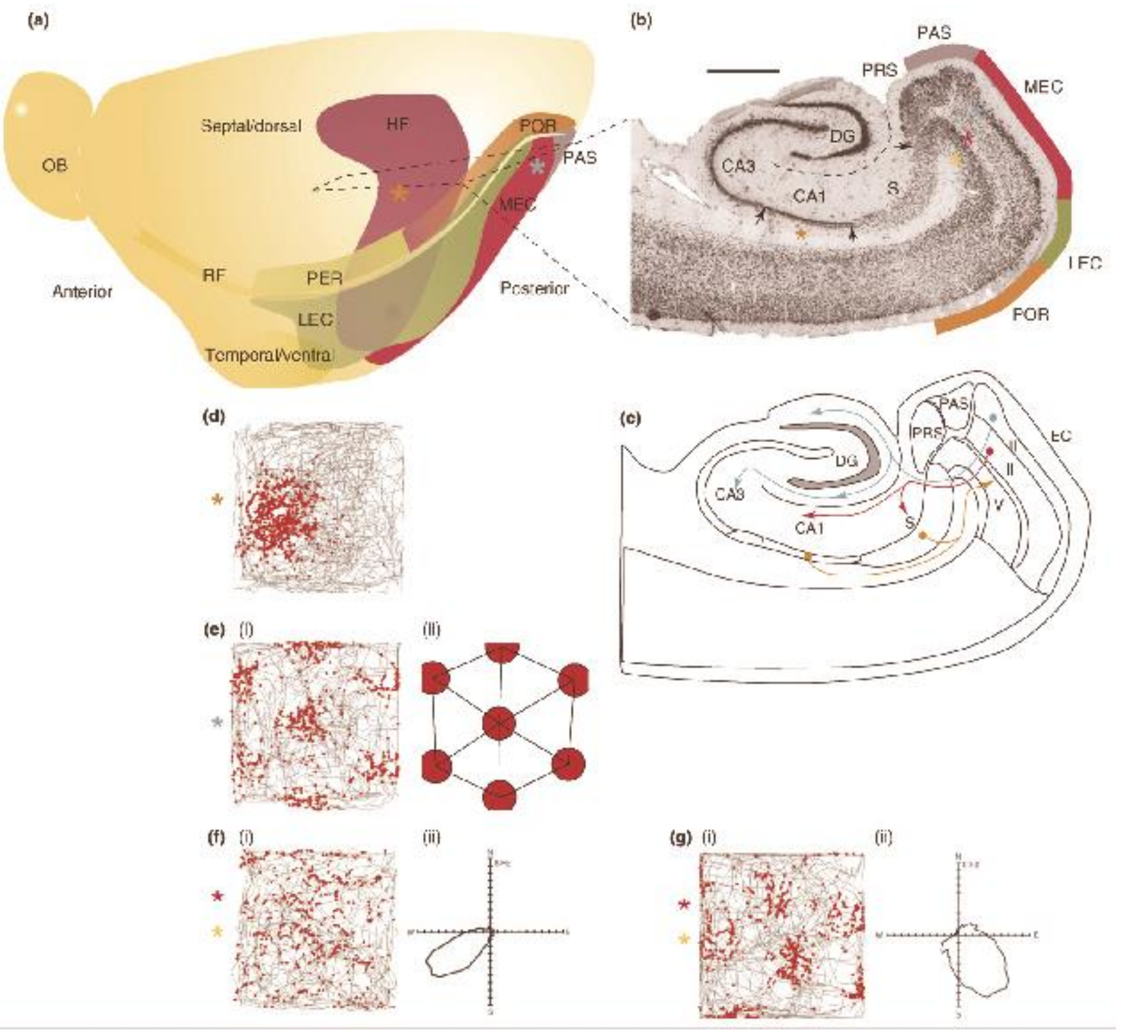


Nonlinear recurrent networks often have activity patterns which behave as fixed point attractors – i.e., they are stable patterns of activity towards which other patterns converge.

These may also be considered as stored memory traces, provided that we can specify the fixed points (preferably via a plausible activity-based learning rule).

Auto-associative function: the dynamics of the network reconstructs the original pattern based on a fragment or a noisy version. This is also known as pattern completion.

The role of the hippocampus in spatial navigation

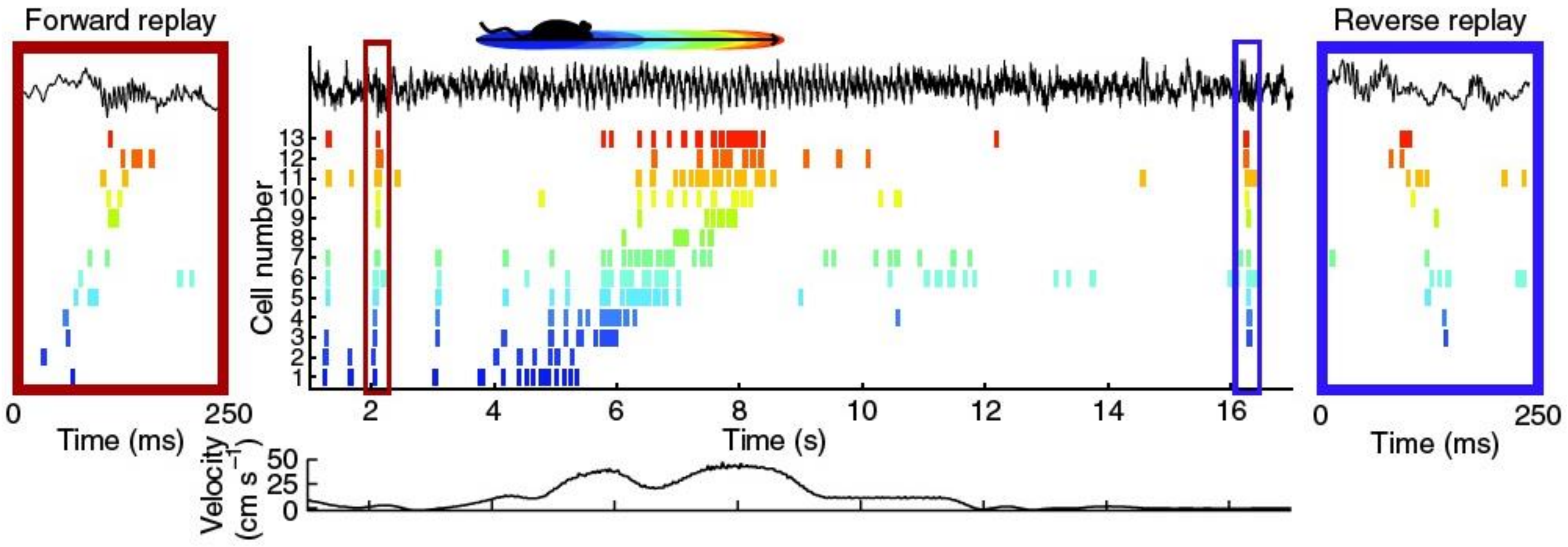


- 50 years ago, place cells were discovered in the hippocampus
- Since then, several other space-related representations (grid cells, head direction cells, boundary cells, etc.) have been discovered in neighboring, connected areas.
- Behaviorally relevant, non-spatial characteristics of the environment are also represented in the hippocampus.
- Hippocampal lesions cause severe deficits in spatial cognition, and specifically in map-based navigation.

These observations led to the cognitive map hypothesis of hippocampal function (O’Keefe and Nadel, 1978).

Sharp wave-ripples and sequence replay

Sharp wave-ripples (SWRs) are autonomously generated bursts of activity in the hippocampus, which are prevalent during awake immobility and slow-wave sleep, and strongly implicated in the formation of long-term memory.



Place cell activation sequences are replayed during SWRs.

Goals

- Investigate, using experimental, theoretical and simulation tools, the mechanisms responsible for the generation of sharp wave-ripples (SWRs) and other dynamic states in area CA3 of the hippocampus in vitro
- See whether these mechanisms, combined with appropriate models of exploration and learning, might also explain the replay of place cell sequences observed during SWRs in vivo
- Understand the relationship between weight structure, the representation of sequences, and global characteristics of the dynamics such as mean rates and population oscillations

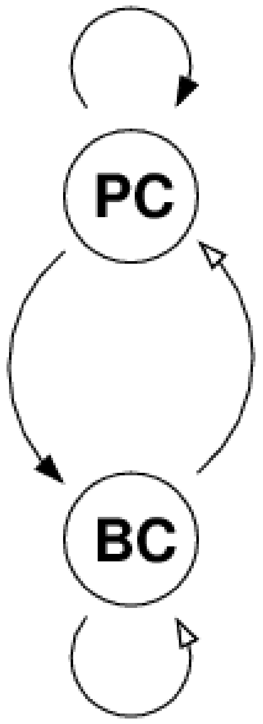
CA3 network model

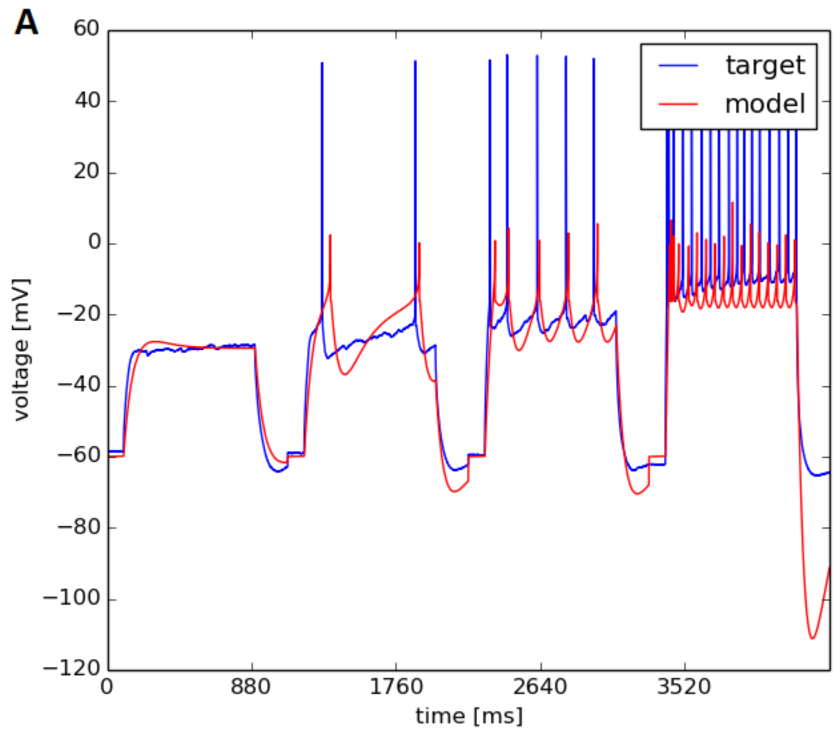
According to in vitro data, pyramidal cells (PCs) and fast-spiking basket cells (BCs) are the key players in SWR generation.

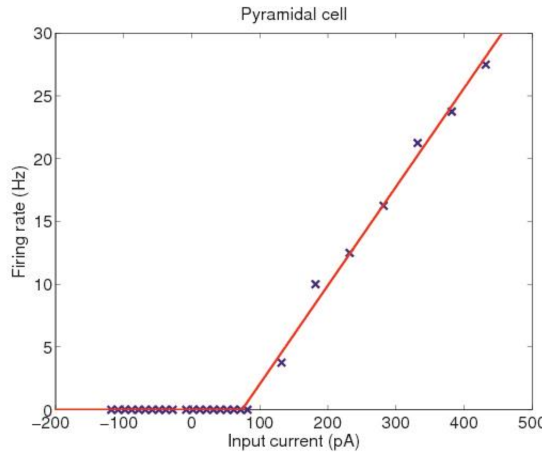
To model area CA3 in the slice, we built spiking networks of 8000 PCs and 150 BCs (some versions also contained 100 "slow interneurons"), taking into account data from

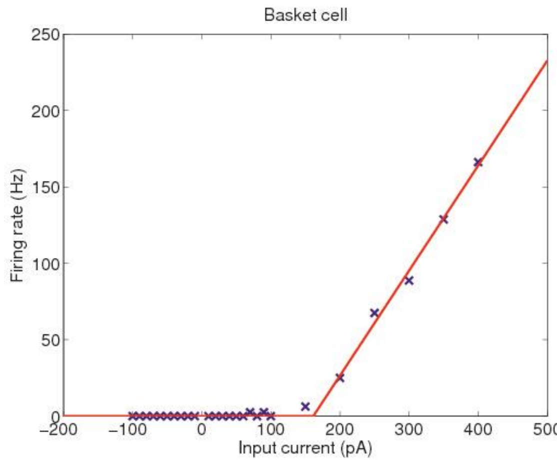
- anatomy (connectivity)
- single cell physiology (response to step currents)
- synaptic physiology

We used our “Neuroptimus" software tool to fit adaptive exponential integrate-and fire (AdExpIF) models to current clamp recordings from PCs and BCs.

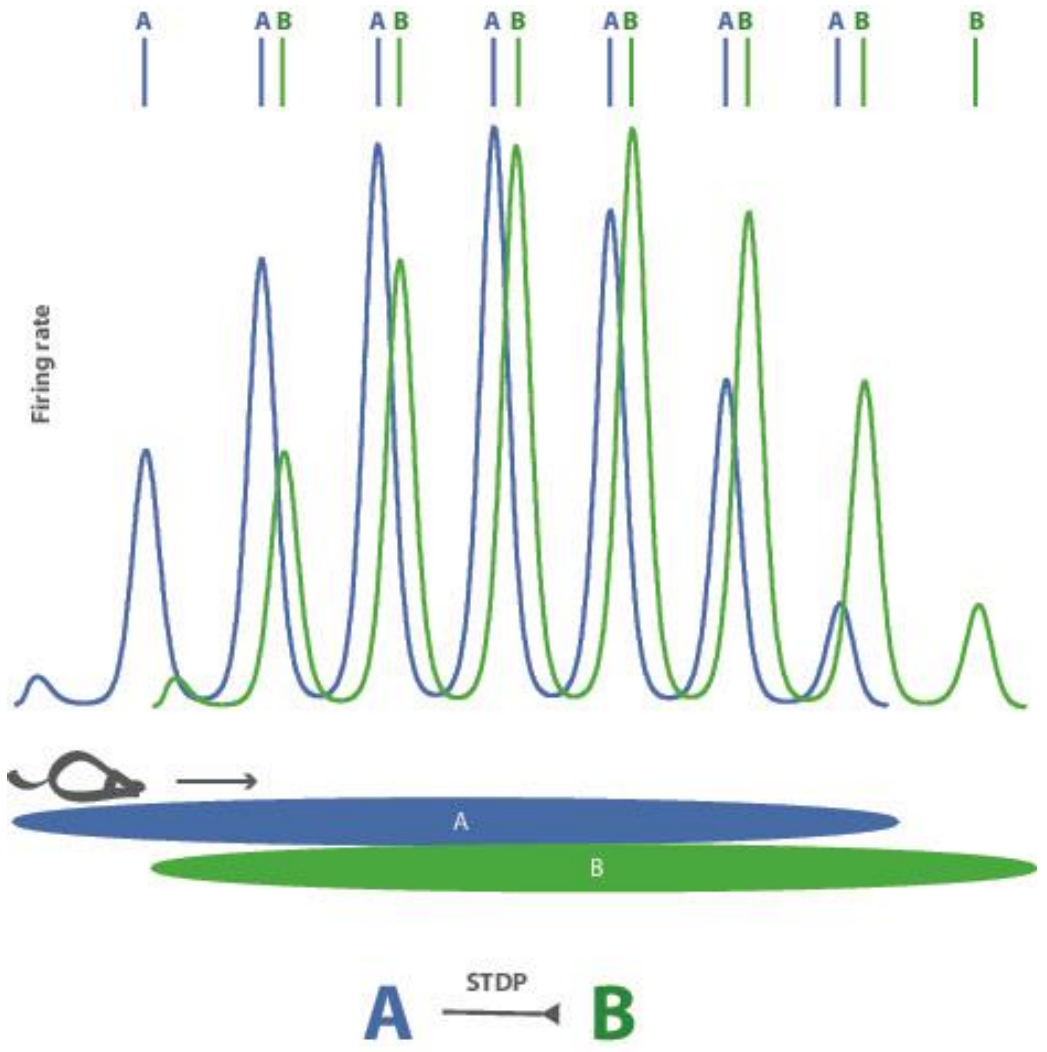




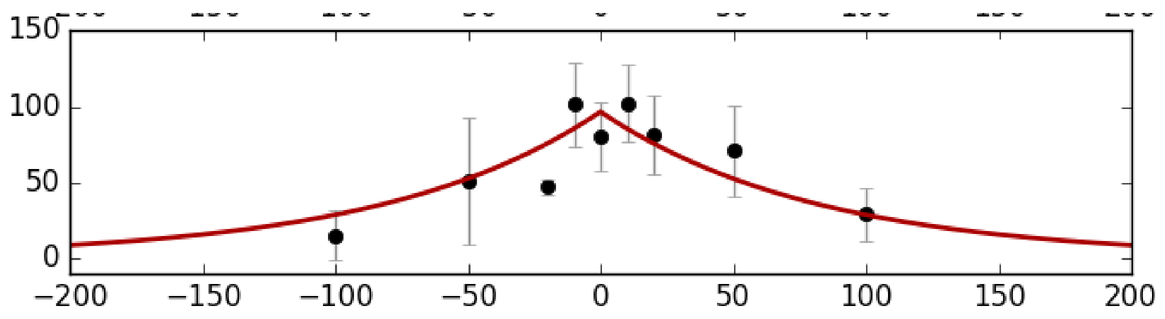




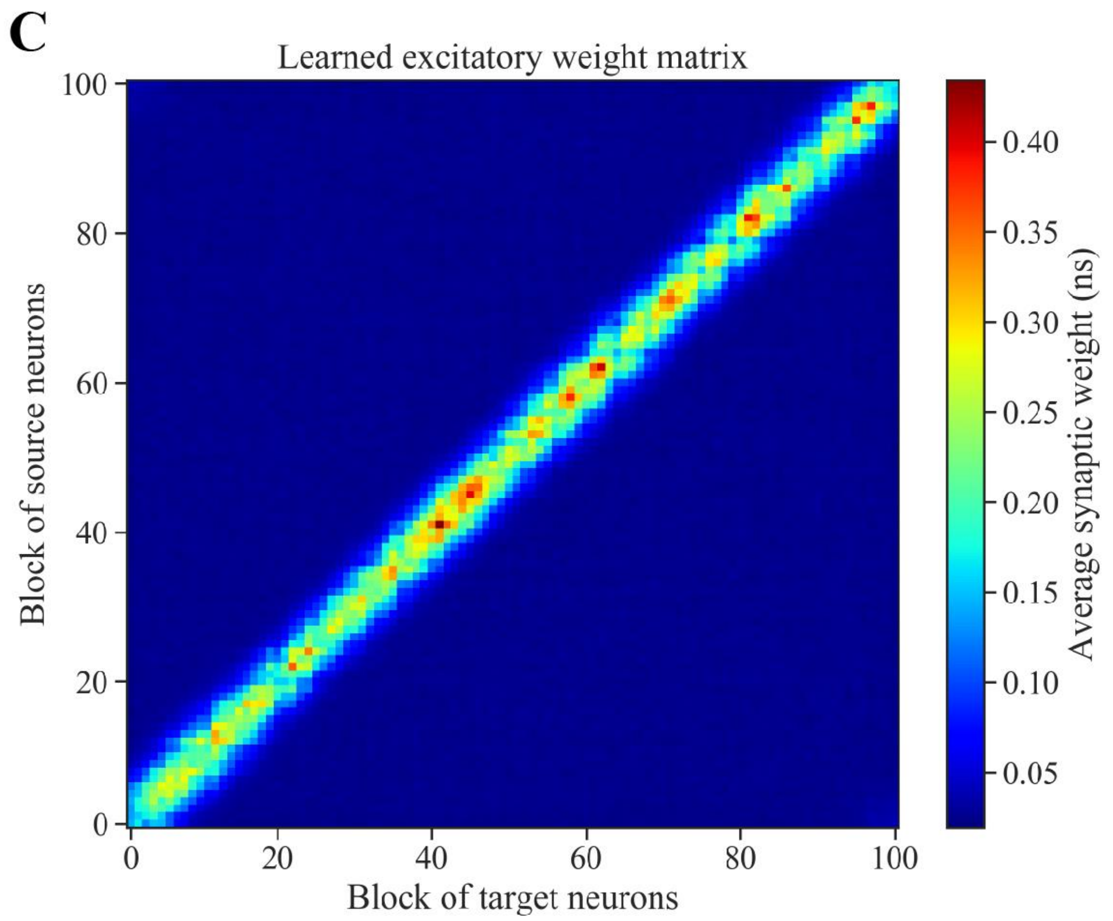
Learning of recurrent weight structure during exploration…



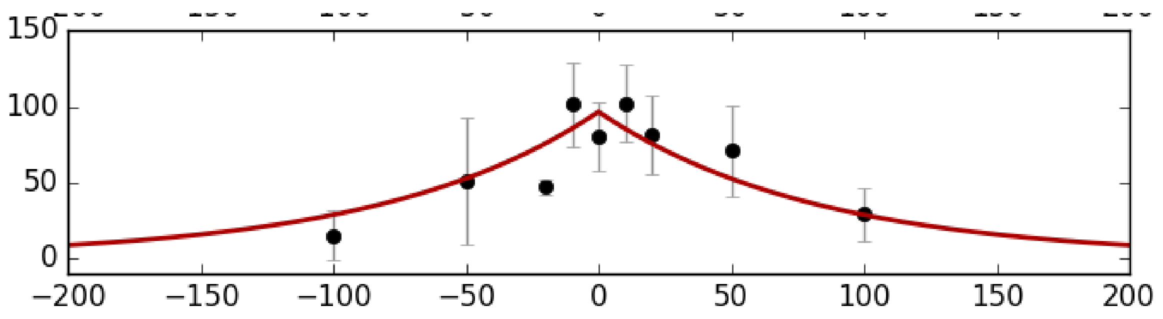
We model the activity of a population of place cells with overlapping place fields during exploration, using location and theta phase-modulated inhomogeneous Poisson processes (phase precession), and apply the symmetric STDP rule measured by Mishra et al. (2016).



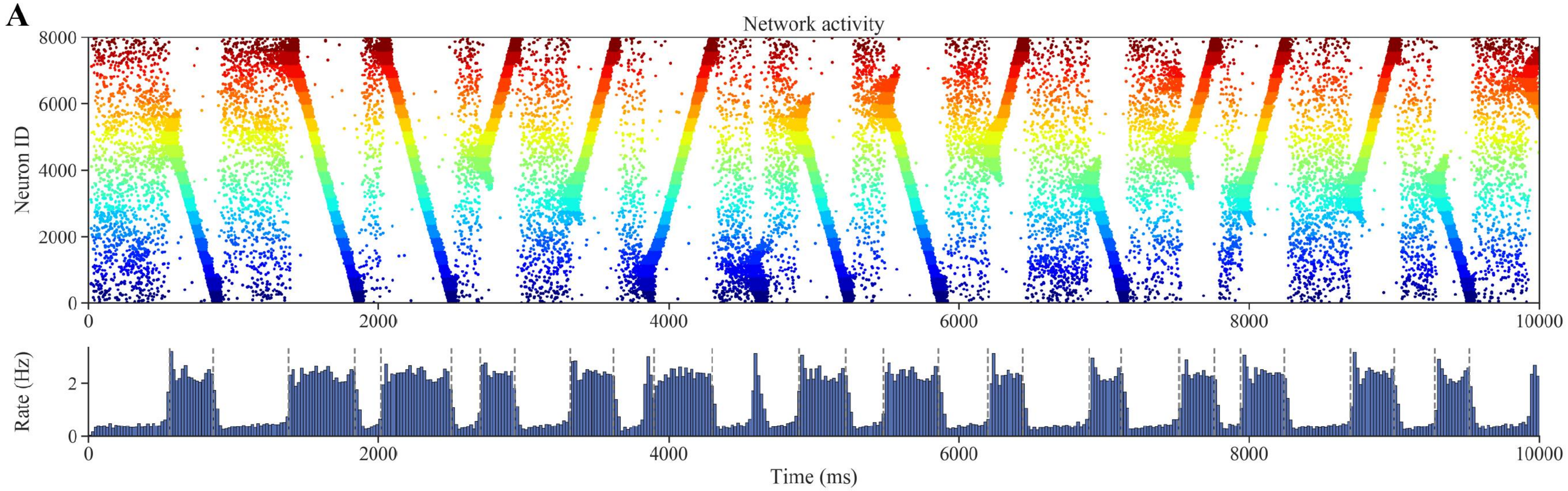
Learning of recurrent weight structure during exploration…



We model the activity of a population of place cells with overlapping place fields during exploration, using location and theta phase-modulated inhomogeneous Poisson processes (phase precession), and apply the symmetric STDP rule measured by Mishra et al. (2016).



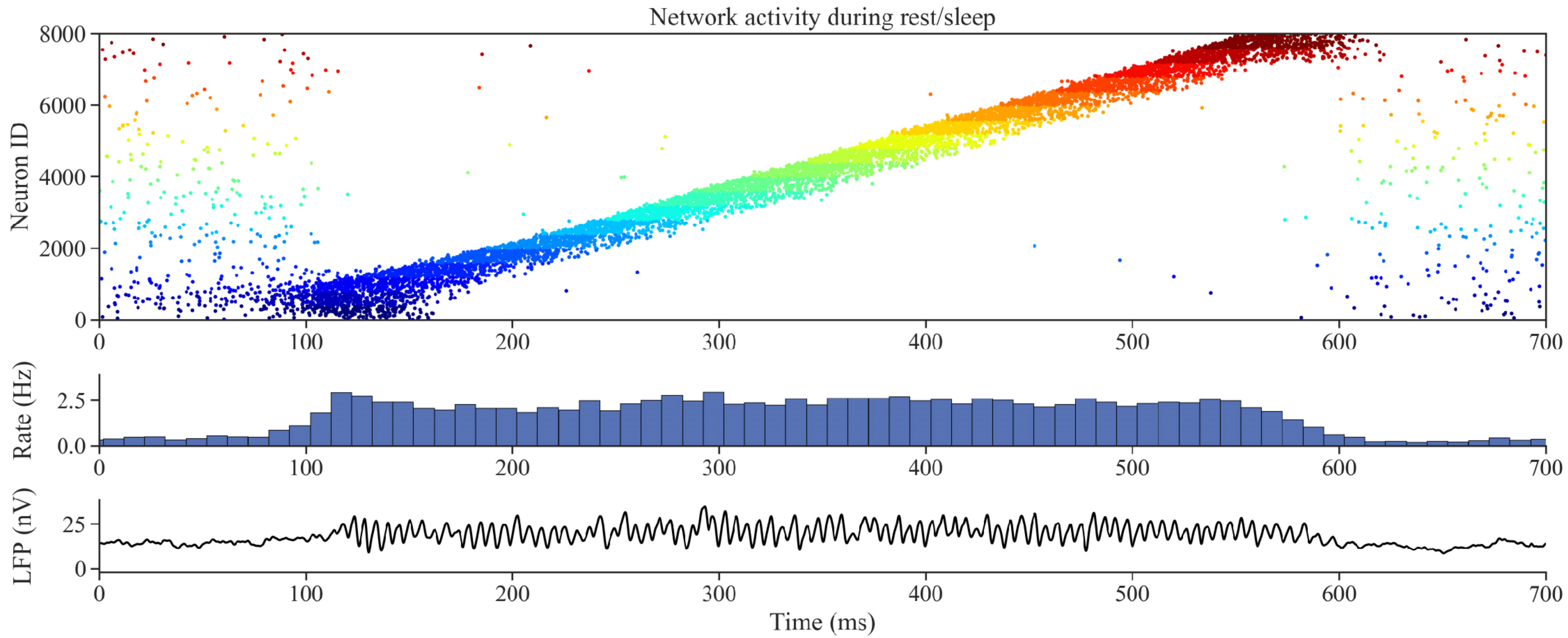
Learning of recurrent weight structure during exploration supports the replay of place cell sequences during SWRs



Replay can occur in both directions, and it can be cued to start from specific locations (not shown) – this may explain the prevalence of forward replay when planning a route, and backward replay at the goal location.

Replay events are accompanied by ripple oscillations

D



Conclusions (SWRs and replay)

- A simple network model of the CA3 area, where most parameters are based on in vitro data but recurrent excitatory weights are learned using STDP during simulated exploration, produces sharp wave-like activity with ripple oscillations and spontaneous replay of learned sequences.
- Manipulations of the weight matrix demonstrate that the distribution of synaptic weights is neither necessary nor sufficient for the observed physiological population activity of the original network.
- Embedded convergent paths in the weight matrix enable robust sequential activity at high (physiological) rates, which in turn leads to the emergence of fast (ripple) oscillations.
- A symmetric learning rule (combined with cellular adaptation) can explain the bidirectional replay of activity sequences.
- The fine structure of recurrent excitation not only enables coding, but also has a major effect on global (average) network dynamics.
- Models constrained by data can reveal principles of dynamics and computation in the brain.

Acknowledgements

Sára Sáray
Máté Mohácsi
Luca Tar
Gábor Farkas

Péter Friedrich
Márk Török
András Ecker
Eszter Vértes
Bence Bagi
Orsolya Németh

Dániel Schlingloff
Zsolt Kohus
Rita Karlócai
Péter Berki

Judit Makara
Attila Gulyás
Norbert Hájos
Tamás Freund

István Miklós

Eilif Muller
Armando Romani
Christian Rössert
Werner Van Geit

Andrew Davison
Shailesh Appukuttan

Michele Migliore
Rosanna Migliore
Paola Vitale
Carmen Lupascu
Luca Bologna

Thank you for your attention!

Questions?