

Summary of example:

In this example we estimate some basic synaptic parameters based on a simulated voltage clamp experiment with synaptic input.

The target data (iclamp.dat) consist of the recorded clamp current (in units of nA) from a virtual voltage clamp electrode inserted into a single-compartment model, which contains Hodgkin-Huxley-

type Na⁺ and K⁺ conductances, plus a conductance-based synapse with a double exponential time course (rise time: 0.3 ms, decay time: 3 ms, maximal conductance: 10 nS, delay: 0 ms). The data file

also contains time.

The model neuron receives through the synapse a spike train input, which consists of 4 spikes at regular 100 ms intervals starting at 100 ms. The full length of the recording is 500 ms, the sampling rate is 40 kHz.

The model file (simple_hh_syn_vclamp_toopt2.hoc) contains the neuronal model (including the synapse), the spike generator (NetStim) object which generates the input, and a NetCon object which

connects the input to the cell.

As we need to set the parameters of the synapse and those of the NetCon, and Optimizer cannot discover these parameters automatically, we use a simple user function (ufun.txt) to adjust the parameters (maximal conductance (in microsiemens), synaptic delay, rise time constant, decay time constant (all in milliseconds)).

We need voltage clamp at a constant level (-70 mV); one way to accomplish this is to use a step protocol in voltage clamp with a single amplitude of -70 mV (and arbitrary delay and duration), and an initial voltage of -70 mV.

The optimization should be done using the mean square error cost function. Evolutionary optimization for 100 generations with a population of 100 restores the original parameters with high precision.

Step-by-step instructions to run the example from the Optimizer GUI:

Run „python3 optimizer.py -g” to start the GUI

Neuroptimus

Menu

Target data

Model

Settings

Fitness

Run

Results

Statistics

Data file

rogate/iclamp_new_eventdt.dat

Browse...

☒ Contains time

Voltage trace ▾

Base directory

w_test_files/VClamp_surrogate

Browse...

Voltage trace

iclamp_new_eventdt.dat

Number of traces

1

Units

mV ▾

Length of traces (ms)

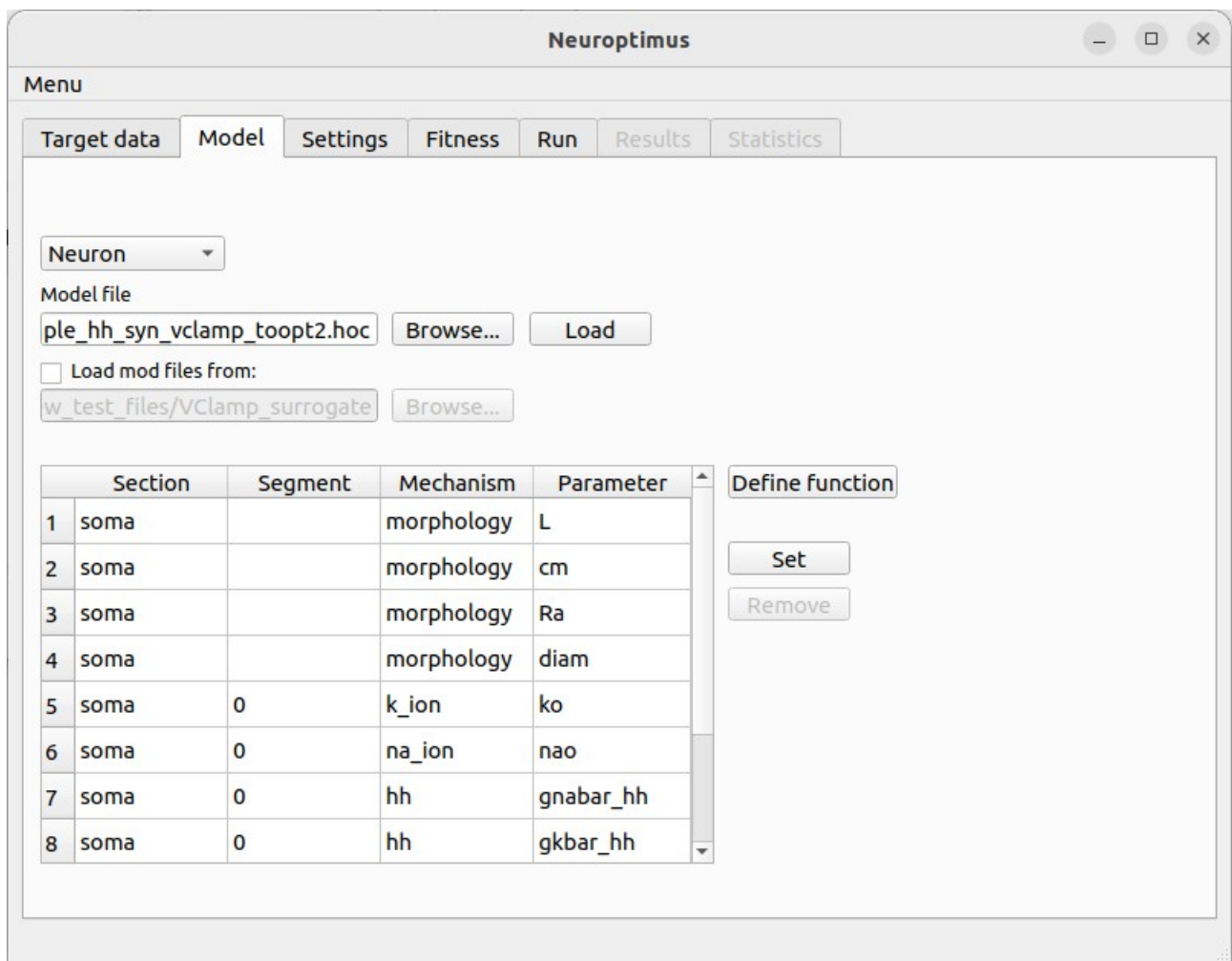
500

Sampling frequency (Hz)

40000.0

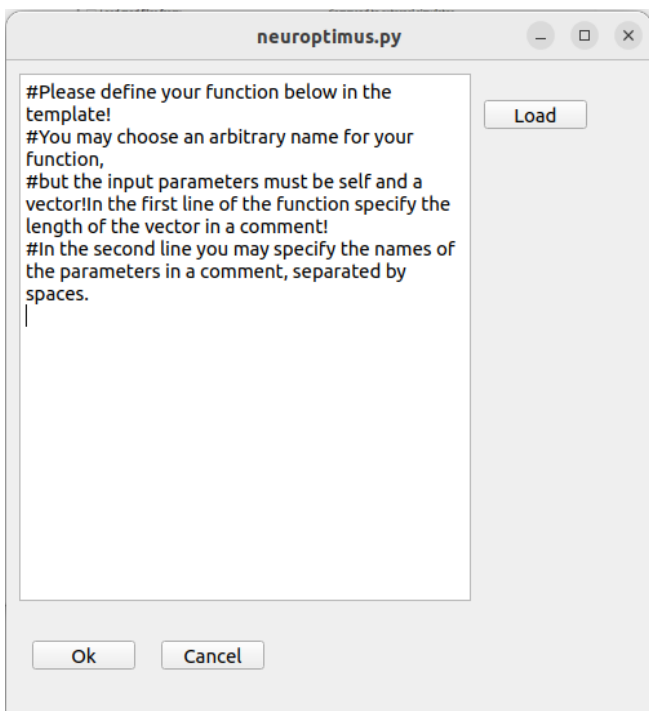
Load data

At 'Data File' load the target data, at 'Base Directory' choose the directory where you want to save the results. Fill out all the cells and press 'Load data'.Go on by pressing the model tab.



Browse to the model file and load the model.

Press the 'Define Function' button to load the user defined function:



Press 'Ok', then go on by pressing the settings tab.

Neuroptimus

Menu

Target data

Model

Settings

Fitness

Run

Results

Statistics

Stimulation protocol

VClamp

Stimulus Type

Step Protocol

Amplitude(s)

Delay (ms)

1

Duration (ms)

1

Section

soma

Position inside section

0.5

Parameter to record

i

Section

soma

Position inside section

0.5

Initial voltage (mV)

-70

tstop (ms)

500.0

Time step

0.025

Fill in all the cells. Press 'Amplitude(s)' to open the 'Set Amplitude(s) window.

neuroptimus.py

Number of stimuli:

1

Create

Amplitude (mV)

1 -70

Accept

Go on by pressing the fitness tab.

The screenshot shows the Neuroptimus software window. The 'Menu' bar at the top contains tabs for 'Target data', 'Model', 'Settings', 'Fitness', 'Run', 'Results', and 'Statistics'. The 'Fitness' tab is currently selected. Above the table is a 'Normalize' button. The table has two columns: 'Fitness functions' and 'Weights'. It lists 10 fitness functions, with 'MSE' selected (weight 1.0) and the others set to 0. To the right of the table are two input fields: 'Spike detection tresh. (mV)' with a value of 0.0, and 'Spike window (ms)' with a value of 1.0.

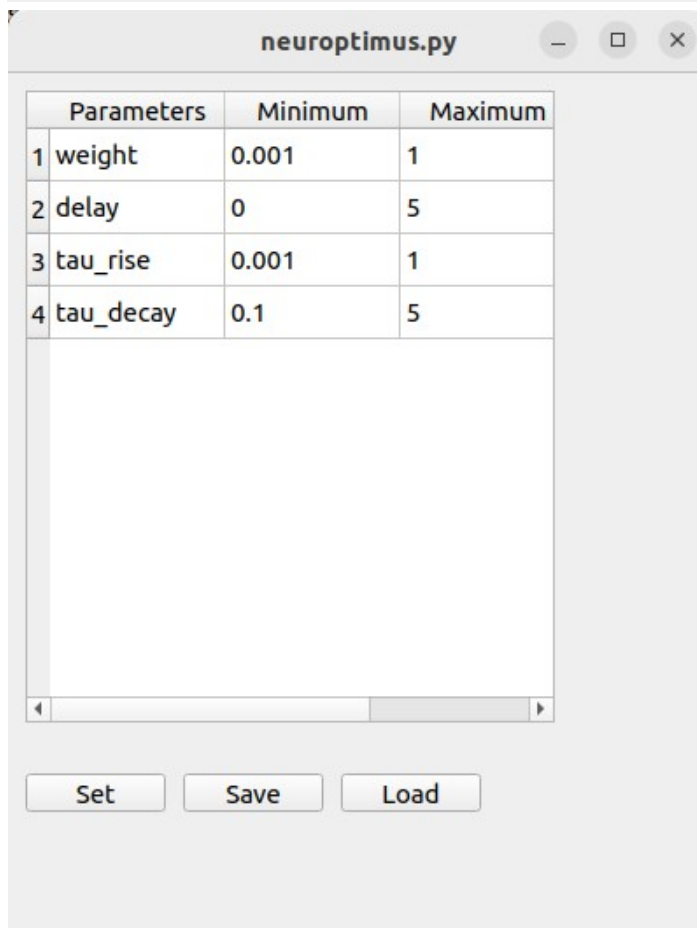
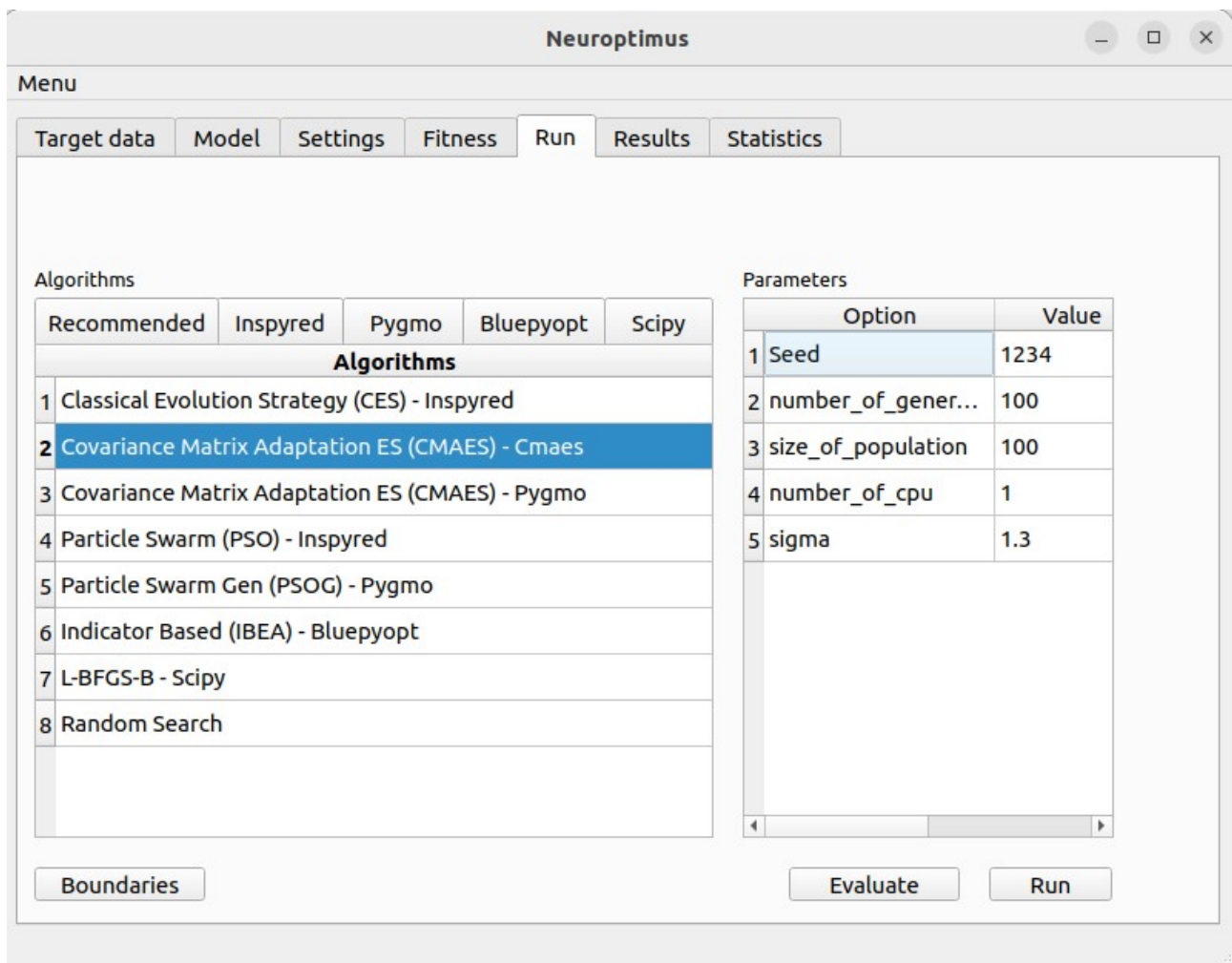
	Fitness functions	Weights
1	MSE	1.0
2	MSE (excl. spikes)	0
3	Spike count	0
4	Spike count (stim.)	0
5	ISI differences	0
6	Latency to 1st spike	0
7	AP amplitude	0
8	AHP depth	0
9	AP width	0
10	Derivative difference	0

Normalize

Spike detection tresh. (mV)
0.0

Spike window (ms)
1.0

Choose fitness function(s) by clicking on them in the first column, and define their weights on the second column. Go on by pressing the run tab.



Select an algorithm, and press the 'Boundaries' button to define the boundaries of the parameters to be optimized:

Press 'Set'.

Start the optimization pressing the 'Run' button.