**Summary of example:**

The input data file (EC_2_Ma_2008_exp_data.txt) contains the relative values of 2 peak EPSP traces, corresponding to the response of a layer II entorhinal cortical neuron to high-frequency (tetanic) stimulation (HFS: 100 Hz, 1 s), in control case and the case of PKA inhibition. The traces are 32 minutes long, the first data point is at 7200000 ms (120 minutes) which is the time point of the stimulation. The peak EPSP time courses were sampled in 2-minute (120000 ms) intervals.

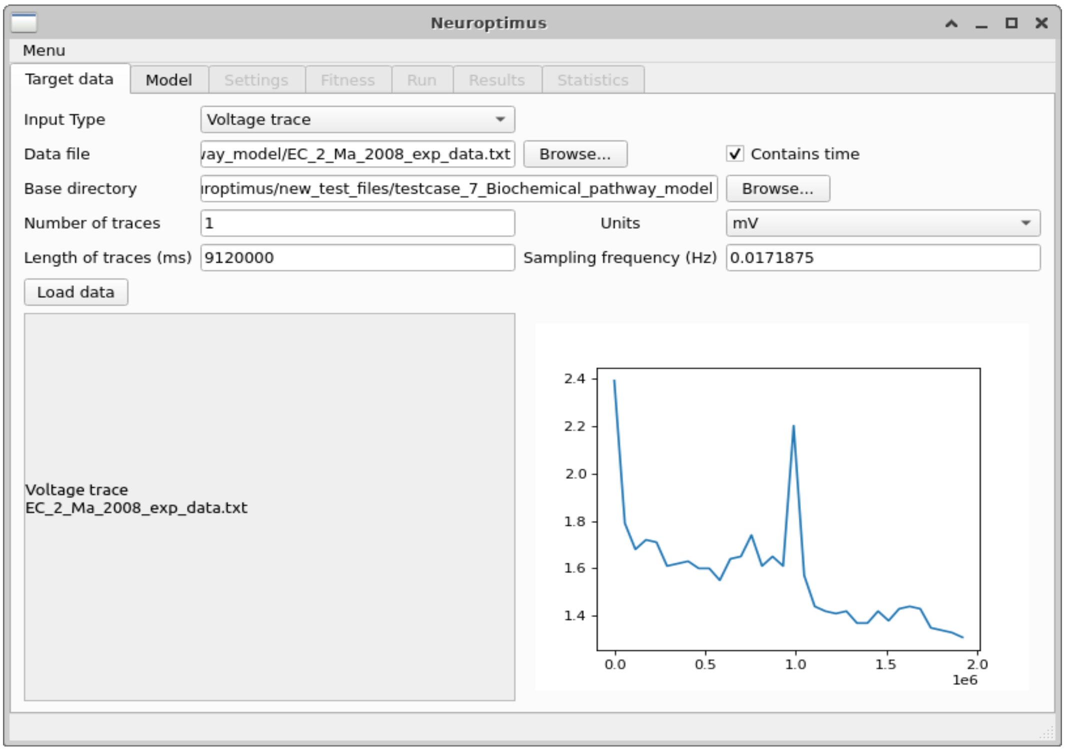
The model to be optimized is a detailed, subcellular biochemical signaling pathway model, which is implemented in Python (model_file_simulation.py), simulated with the reaction-diffusion (rxd) submodule of the NEURON simulator, and handled as a black box by Neuroptimus. The model has the following 12 parameters that are optimized by the software:
Caflux,
Lflux,
Gluflux,
GluR1_ratio.
IC_MGluRM1RGqPLC (IC_PLC),
IC_RGsAC1AC8 (IC_AC),
IC_CaMCK,
IC_NCX,
IC_PKC,
IC_PKA,
IC_PP1PP2B (IC_PP),
IC_PDE1PDE4 (IC_PDE)

The run.py script sets the parameter values, executes the model simulations, and saves the results of the simulations in a trace.dat file (containing the relative peak EPSP traces), which are then used by Neuroptimus to compute the corresponding fitness value using mean-squared error as a cost function.

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

Run „python3 neuroptimus.py -g" to start the GUI

On the 'Target data' tab, at 'Data file' load the target data, and at 'Base directory' choose the directory where you want to save the results. Tick off the 'Contains time' checkbox and press 'Load data' to obtain the target date trace. Go on by pressing the 'Model' tab.



Input Type: Voltage trace

Data file: ｪay_model/EC_2_Ma_2008_exp_data.txt

☑ Contains time

Base directory: ıroptimus/new_test_files/testcase_7_Biochemical_pathway_model

Number of traces: 1

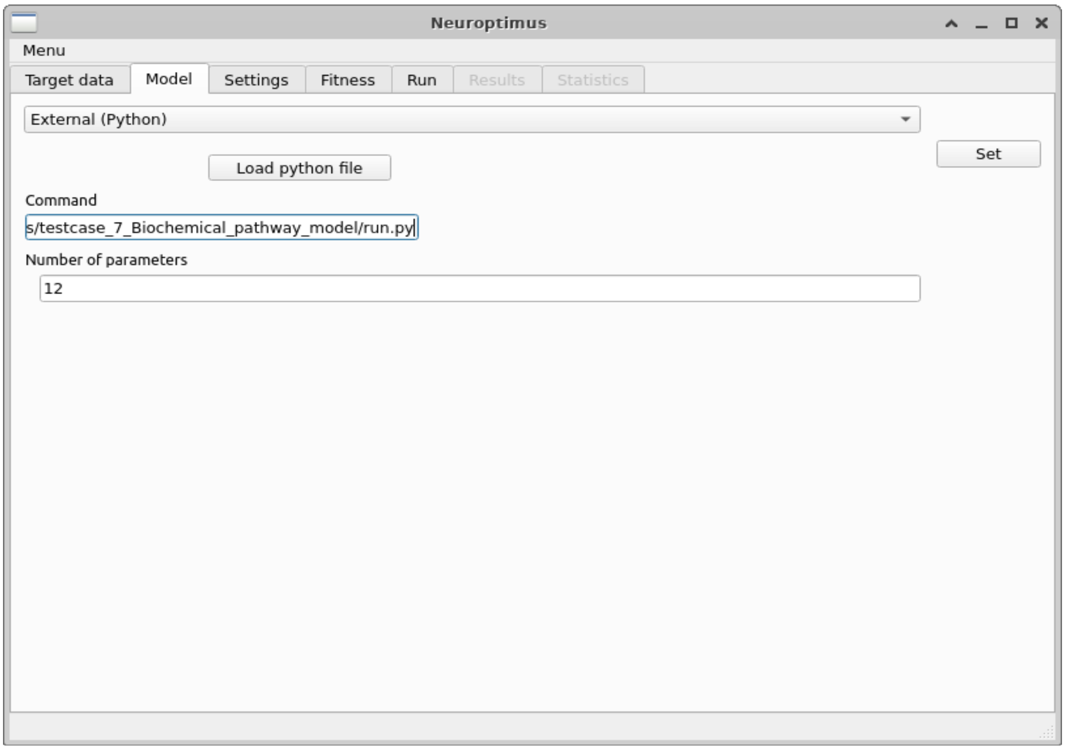
Units: mV

Length of traces (ms): 9120000

Sampling frequency (Hz): 0.0171875

Voltage trace
EC_2_Ma_2008_exp_data.txt

For model type choose 'External (Python)' and load the Python file, or fill out the 'Command' box such: "python3 /FULL/PATH/TO/MODEL/SCRIPT.py 12", where /FULL/PATH/TO/MODEL/SCRIPT.py points to the location of the Python script which runs the simulations (for example: "python3 /PATH_TO_SIMULATION_FILE/neuroptimus/new_test_files/testcase_7_Biochemical_pathway_model/run.py 12", where 12 is the number of parameters subject to optimization). Press 'Set' and go on by pressing the 'Fitness' tab.
(On the 'Settings' tab, settings of the simulation, stimulation, and recording can be adjusted, but a basic configuration is set automatically, which is not necessary to modify.)

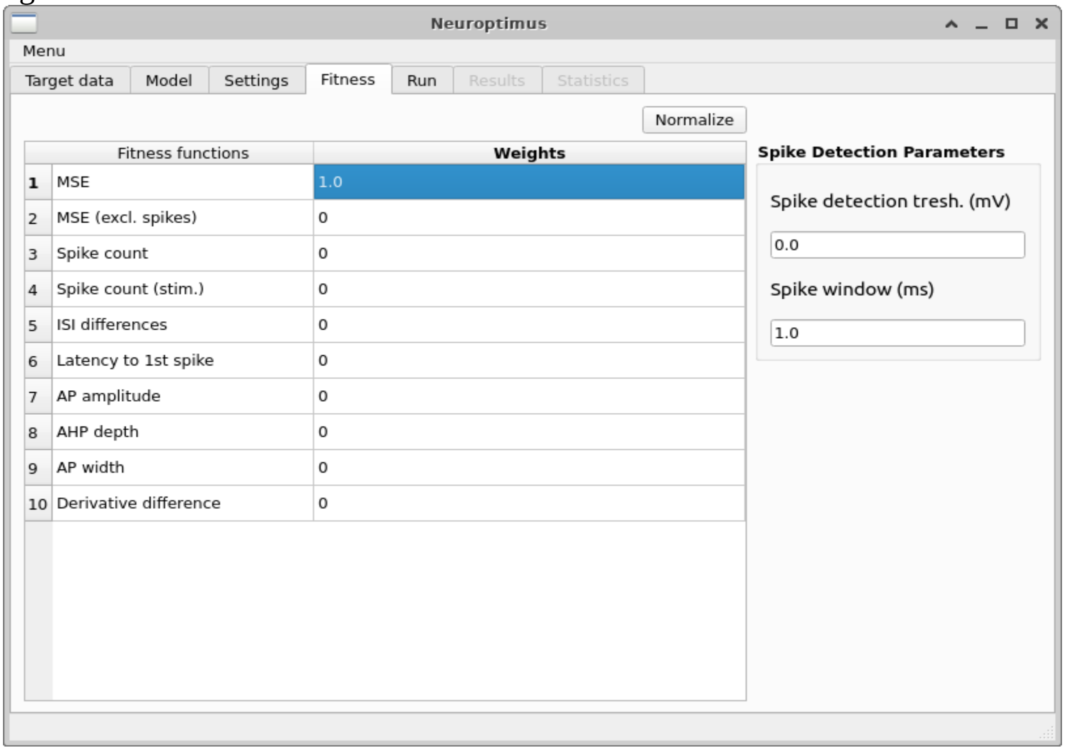


External (Python)

Command: s/testcase_7_Biochemical_pathway_model/run.py

Number of parameters: 12

On the ‘Fitness’ tab choose the ‘MSE’ fitness function, and set its weight to 1. Go on by pressing the ‘Run’ Tab.



|  | 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 |

**Spike Detection Parameters**

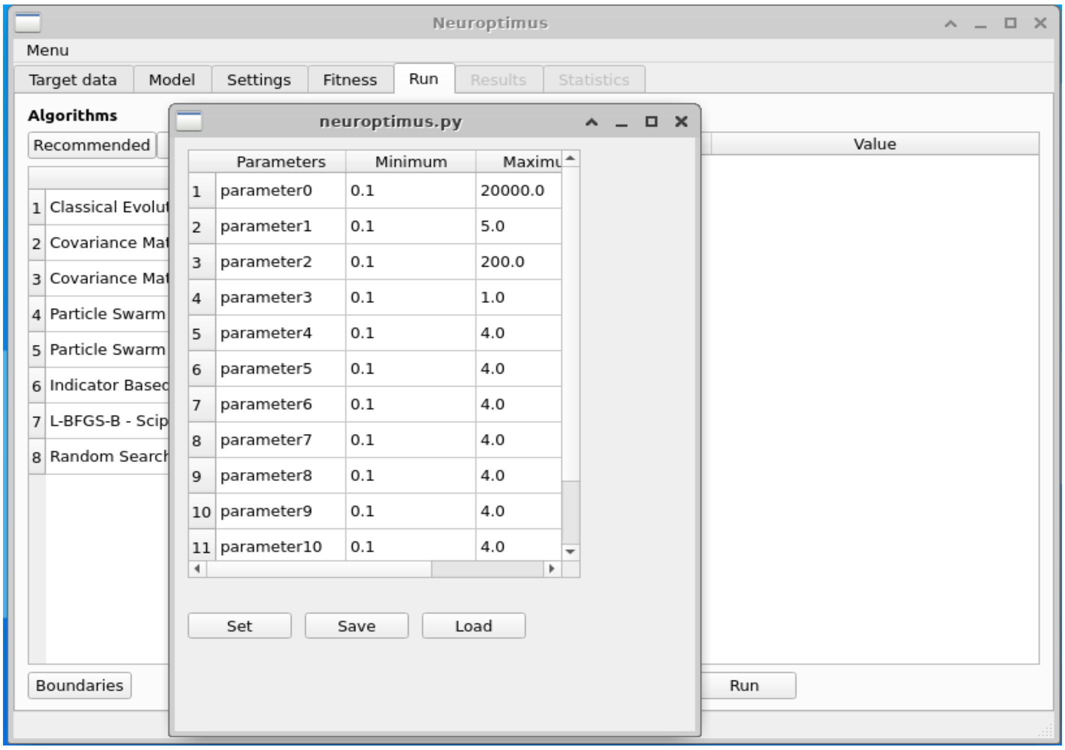
Spike detection tresh. (mV)

0.0

Spike window (ms)

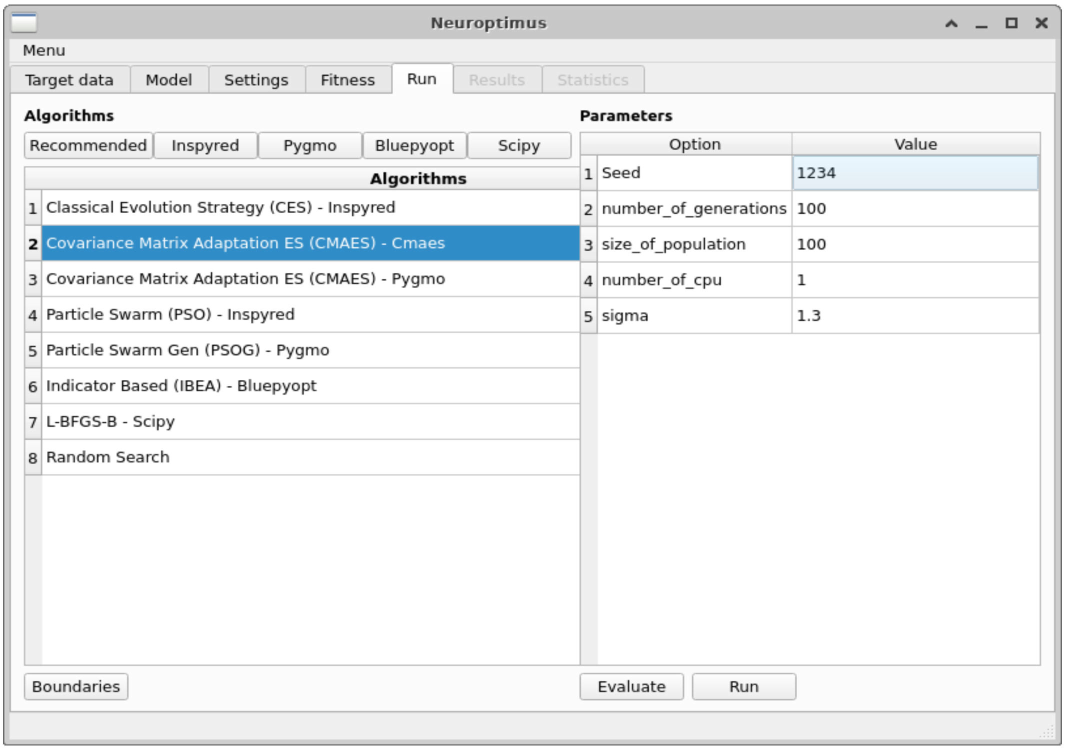
1.0

Press the ’Boundaries’ button at the left bottom corner to define the boundaries of the parameters to be optimized. Boundaries can also be loaded from a file (e.g., the file ‘boundaries.txt’ included with this example) by pressing the ‘Load’ button. After setting the parameter boundaries, press ’Set’.



|  | Parameters | Minimum | Maximu |
|---|---|---|---|
| 1 | parameter0 | 0.1 | 20000.0 |
| 2 | parameter1 | 0.1 | 5.0 |
| 3 | parameter2 | 0.1 | 200.0 |
| 4 | parameter3 | 0.1 | 1.0 |
| 5 | parameter4 | 0.1 | 4.0 |
| 6 | parameter5 | 0.1 | 4.0 |
| 7 | parameter6 | 0.1 | 4.0 |
| 8 | parameter7 | 0.1 | 4.0 |
| 9 | parameter8 | 0.1 | 4.0 |
| 10 | parameter9 | 0.1 | 4.0 |
| 11 | parameter10 | 0.1 | 4.0 |

Thereafter, select an algorithm, and set its parameters. By pressing the ‘Run’ button, the optimization can be launched.



| | Algorithms |
|---|---|
| 1 | Classical Evolution Strategy (CES) - Inspyred |
| 2 | Covariance Matrix Adaptation ES (CMAES) - Cmaes |
| 3 | Covariance Matrix Adaptation ES (CMAES) - Pygmo |
| 4 | Particle Swarm (PSO) - Inspyred |
| 5 | Particle Swarm Gen (PSOG) - Pygmo |
| 6 | Indicator Based (IBEA) - Bluepyopt |
| 7 | L-BFGS-B - Scipy |
| 8 | Random Search |

| | Option | Value |
|---|---|---|
| 1 | Seed | 1234 |
| 2 | number_of_generations | 100 |
| 3 | size_of_population | 100 |
| 4 | number_of_cpu | 1 |
| 5 | sigma | 1.3 |

A configuration file (e.g., ‘testcase_7_Biochemical_pathway_model_settings.json’) is created by running an optimization through the GUI. This file contains the settings adjusted throughout the GUI and can be used to run optimizations later without using the GUI. The settings can be changed easily by editing the configuration file which provides flexibility of running optimizations.