Human Brain Project

TVB-multiscale:

interfacing TVB with NEST (and ANNarchy) spiking networks for multiscale co-simulation

D Perdikis, L. Domide, J. Mersmann, M. Schirner, P. Ritter
Brain Simulation Section, Department of Neurology,
Charité—Universitätsmedizin Berlin

CHARITÉ UNIVERSITÄTSMEDIZIN BERLIN

BRAIN SIMULATION SECTION

THEVIRTUALBRAIN.

Co-funded by the European Union

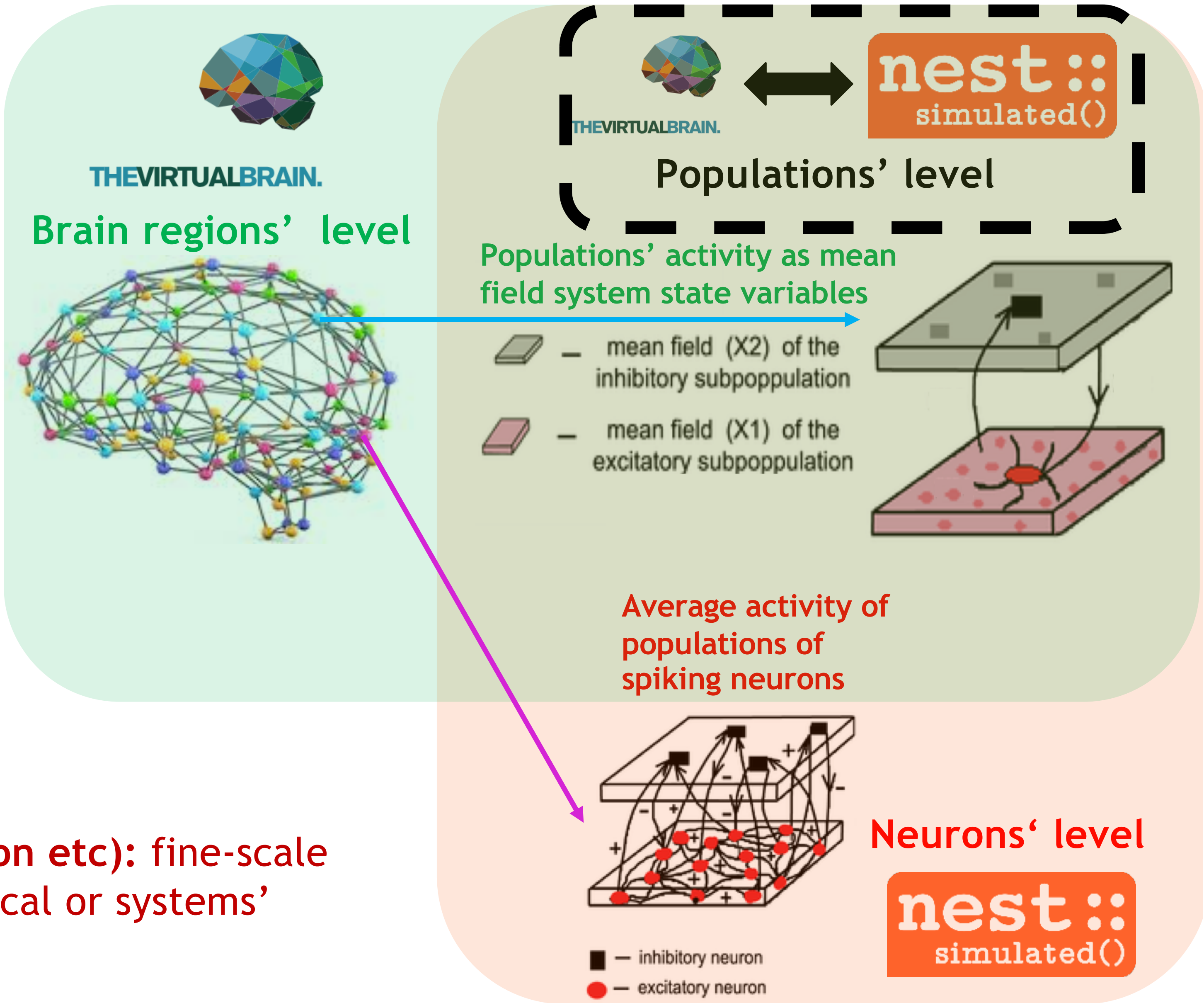
Human Brain Project

Motivation

TVB: large-scale simulation linking brain anatomical data ((d)MRI, CT etc) to neuroimaging data, by generating virtual neural source activity.

⬍

Spiking simulators (NEST, Neuron etc): fine-scale simulation for investigation of local or systems' neural mechanisms



Software structure

- ***TVB-multiscale/tvb_multiscale/core:*** extends TVB to perform multiscale co-simulation.
  - spiking_models: region_node, devices, network, and their builders
  - interfaces: tvb_to_spikeNet, spikeNet_to_tvb, and their builders
  - tvb/simulator_builder
  - data_analyzer/spiking_network_analyzer
  - plot, io (read/write to/from .h5 files), utils (utility scripts)
- ***TVB-multiscale/tvb_multiscale/tvb_nest:*** Python interface of TVB with (Py)NEST, for co-simulation of TVB and NEST models.
  - nest_models: region_node, devices, network, their builders, builders of specific models
  - interfaces: tvb_to_nest, nest_to_tvb, their builders, builders of specific models
- ***TVB-multiscale/tvb_multiscale/tvb_annarchy:*** Python interface of TVB with ANNarchy, for co-simulation of TVB and ANNarchy models.
  - annarchy_models: region_node, devices, network, their builders, builders of specific models
  - interfaces: tvb_to_annarchy, annarchy_to_tvb, their builders, builders of specific models
- ***TVB-multiscale/tvb_multiscale/tvb_elephant:***
  - spike_stimulus_builder
  - spiking_network_analyzer
- ***TVB-multiscale/*** docker, examples, tests, docs

https://github.com/the-virtual-brain/tvb-multiscale

https://hub.docker.com/r/thevirtualbrain/tvb-nest3

Default workflow

1. Setup configuration
   (min_delay, integration time resolution etc)
2. Build TVB model and simulator
   (connectivity, model, coupling, integrator, monitors etc)
3. Build Spiking Network model
   (neural populations, stimulation and measuring devices)
4. Build interfaces
5. (Co-)Simulate
6. Gather results, analyze, plot, write to .h5 files

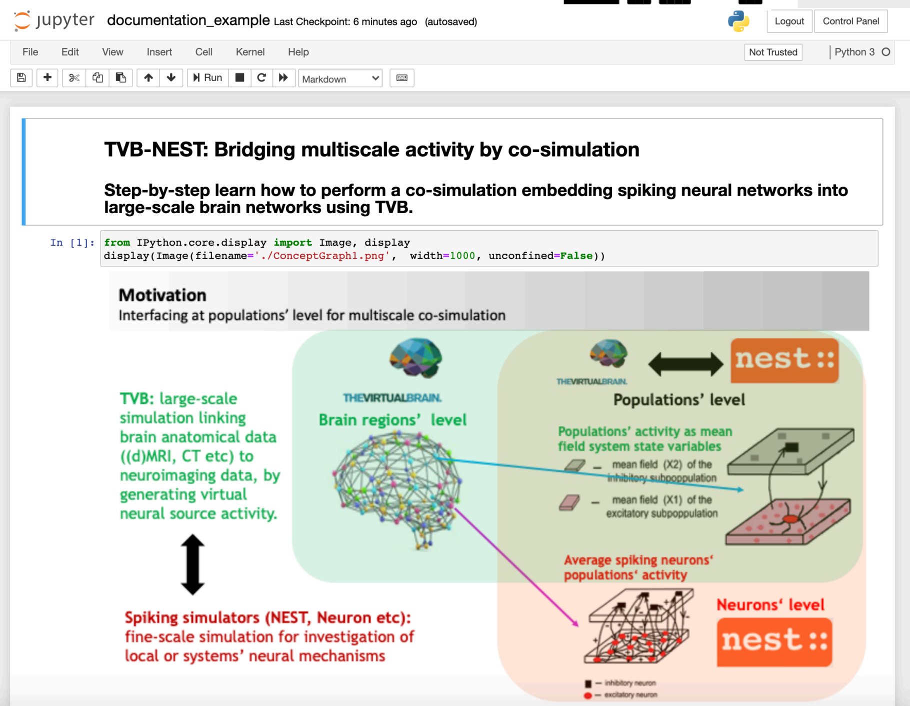
Human Brain Project

EBRAINS collaboratory

Co-Simulation The Virtual Brain Multiscale

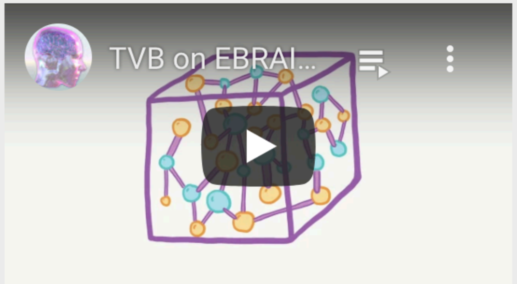
Last modified by Lia Domide on 2020/08/16 14:10

https://tvb-nest.apps.hbp.eu/



https://wiki.ebrains.eu/bin/view/Collabs/the-virtual-brain-multiscale/

TVB Co-Simulation



Multiscale: TVB - NEST

Authors: D. Perdikis, L. Domide, J. Mersmann, M. Schirner, P. Ritter

Human Brain Project

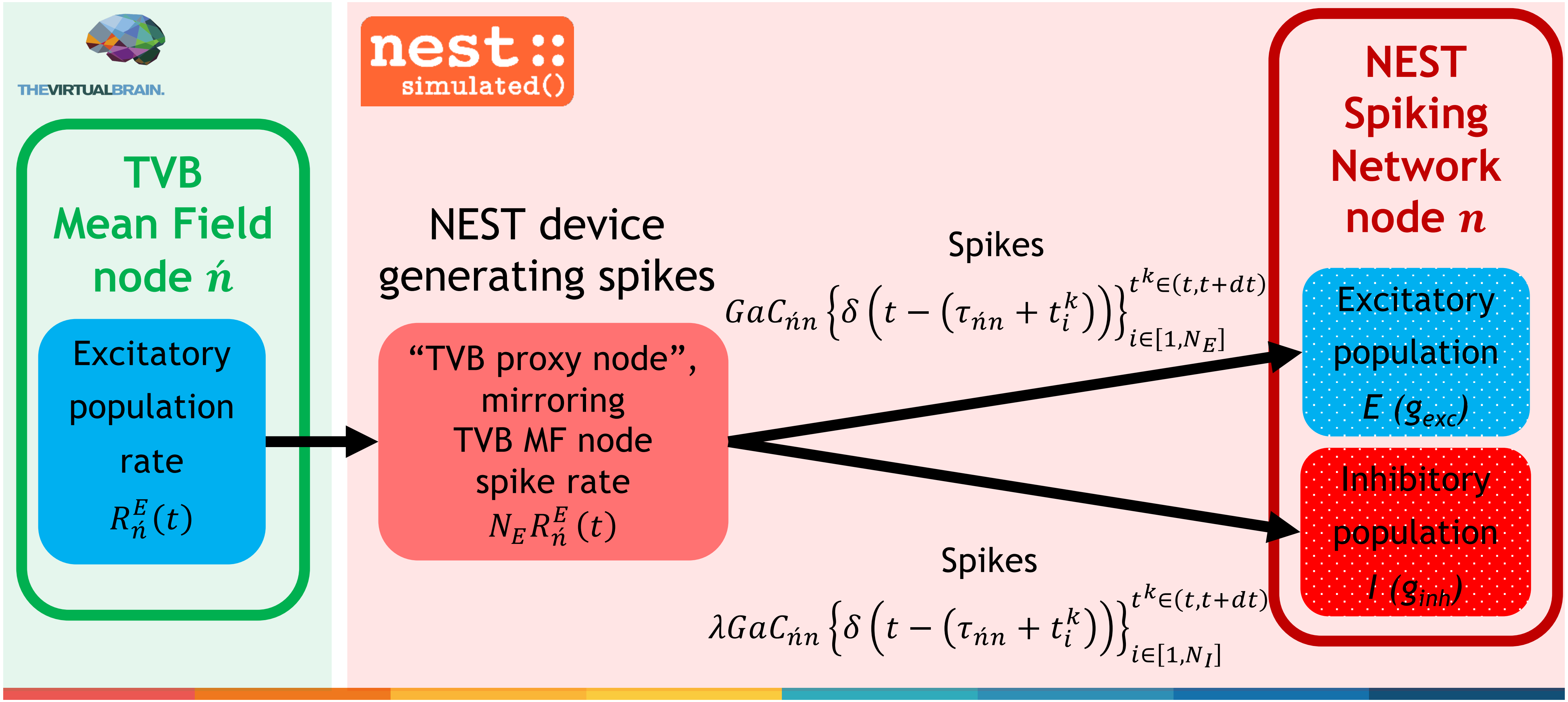
TVB-multiscale on EBRAINS collaboratory

❖ Use case in EBRAINS Collaboratory:

- 1 excitatory (E) and 1 inhibitory (I) population per region node
- Coarse-scale MF nodes in TVB with a reduced Wong-Wang model
- fine-scale spiking networks SP nodes with an ”integrate-and-fire”, (alpha) conductance model (“iaf_cond_alpha”).

Use case

TVB to NEST coupling via TVB "proxy" nodes: spike rate



Use case

Human Brain Project

TVB to NEST coupling via TVB "proxy" nodes: spike rate

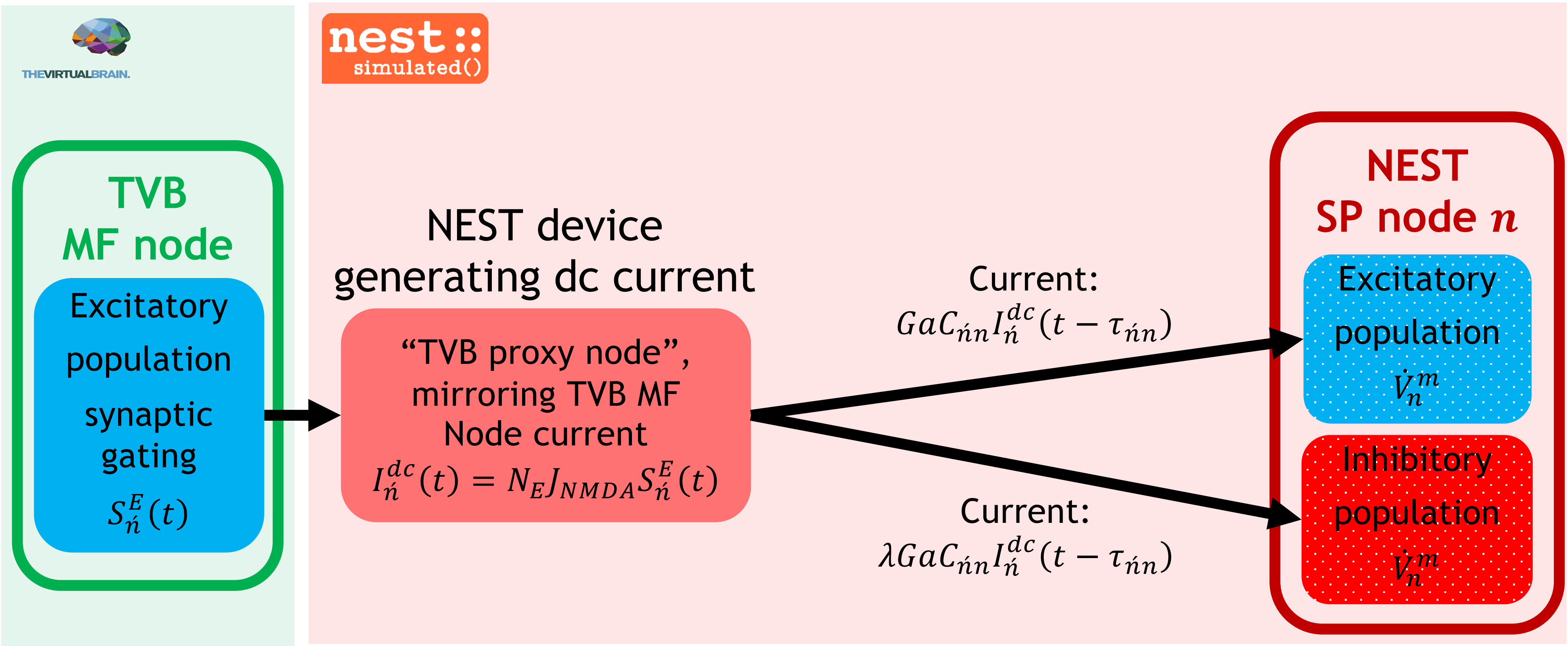
```
tvb_nest_builder.tvb_to_spikeNet_interfaces = [
    {"model": "inhomogeneous_poisson_generator",
     "params": {"allow_offgrid_times": False},
# # ---------Properties potentially set as function handles with args (tvb_node_id=None)---------------------
     "interface_weights": 1.0 * N_E, # Convert mean value to total value
# Applied outside NEST for each interface device
# -------Properties potentially set as function handles with args (tvb_node_id=None, nest_node_id=None)------
    "weights": tvb_weight_fun, "delays": tvb_delay_fun, "receptor_type": 0,
    # ----------------------------------------------------------------------------------------------------
    #             TVB sv -> NEST population
    "connections": {"R_e": ["E"]},
    "source_nodes": None, "target_nodes": None}]  # None means all here

if lamda > 0.0:
    tvb_nest_builder.tvb_to_spikeNet_interfaces.append(
        {"model": "inhomogeneous_poisson_generator",
         "params": {"allow_offgrid_times": False},
    # # ---------Properties potentially set as function handles with args (tvb_node_id=None)-----------------
         "interface_weights": lamda * N_E, # Convert mean value to total value
    # Applied outside NEST for each interface device
    # -------Properties potentially set as function handles with args (tvb_node_id=None, nest_node_id=None)--
        "weights": tvb_weight_fun, "delays": tvb_delay_fun, "receptor_type": 0,
        # ------------------------------------------------------------------------------------------------
        #             TVB sv -> NEST population
        "connections": {"R_e": ["I"]},
        "source_nodes": None, "target_nodes": None
        }
    )
```

Use case

Human Brain Project

TVB to NEST coupling via TVB “proxy” nodes: current



Human Brain Project

Use case

TVB to NEST coupling via TVB "proxy" nodes: current

```
# --For injecting current to NEST neurons via dc generators acting as TVB proxy nodes with TVB delays:--

tvb_nest_builder.tvb_to_spikeNet_interfaces = [
    {"model": "dc_generator", "params": {},
# Properties potentially set as function handles with args (tvb_node_id=None)
#   Applied outside NEST for each interface device
     "interface_weights": 1.0,  # N_E / N_E
# Properties potentially set as function handles with args (tvb_node_id=None, nest_node_id=None)
     "weights": tvb_weight_fun, "delays": tvb_delay_fun,
#                   TVB sv -> NEST population
     "connections": {"S_e": ["E"]},
     "source_nodes": None, "target_nodes": None}]  # None means all here

if lamda > 0.0:
    tvb_nest_builder.tvb_to_spikeNet_interfaces.append(
       {"model": "dc_generator", "params": {},
    # ---------Properties potentially set as function handles with args (tvb_node_id=None)
    #   Applied outside NEST for each interface device
        "interface_weights": lamda*N_E / N_I,
    # Properties potentially set as function handles with args (tvb_node_id=None, nest_node_id=None)
        "weights": tvb_weight_fun, "delays": tvb_delay_fun,
    #                   TVB sv -> NEST population
        "connections": {"S_e": ["I"]},
        "source_nodes": None, "target_nodes": None}
    )
```

Human Brain Project

Use case

TVB to NEST coupling via direct parameter update

THEVIRTUALBRAIN.

SP node n in TVB

Excitatory population

large-scale delayed coupling

$$S_n^{coupl}(t) = a \sum_{\acute{n}} C_{\acute{n}n} S_{\acute{n}}^{E}(t - \tau_{\acute{n}n})$$

nest :: simulated()

NEST

external current parameter

$$I_n^{ext}(t) = N_E G J_{NMDA} S_n^{coupl}(t)$$

$$I_n^{ext}(t) = N_E \lambda G J_{NMDA} S_n^{coupl}(t)$$

NEST SP node n

Excitatory population

$\dot{V}_n^m$

Inhibitory population

$\dot{V}_n^m$

Human Brain Project

Use case

TVB to NEST coupling via direct parameter update

```
tvb_nest_builder.tvb_to_spikeNet_interfaces = [
    {"model": "current",  "parameter": "I_e",
# Properties potentially set as function handles with args (tvb_node_id=None)
     "interface_weights": 1.0,  # N_E / N_E
#                  TVB sv -> NEST population
     "connections": {"S_e": ["E"]},
     "nodes": None}]  # None means all here
if lamda > 0.0:
    # Coupling to inhibitory populations as well (feedforward inhibition):
    tvb_nest_builder.tvb_to_spikeNet_interfaces.append(
    {
        "model": "current", "parameter": "I_e",
# Properties potentially set as function handles with args (tvb_node_id=None)
        "interface_weights": lamda * N_E / N_I,
#                     TVB sv -> NEST population
        "connections": {"S_e": ["I"]},
        "nodes": None}
)
```

Use case

Human Brain Project

NEST to TVB update

THEVIRTUALBRAIN.

nest::simulated()

TVB Mean Field node n

Excitatory population spike rate

$$Rin_n^E(t) = \frac{N_{spikes}^{E\ in\ n}(t)}{dtN_E}$$

Inhibitory population spike rate

$$Rin_n^I(t) = \frac{N_{spikes}^{I\ in\ n}(t)}{dtN_I}$$

NEST spike detector devices

spike count

$$N_{spikes}^{E\ in\ n}(t) = \sum_{i=1}^{N_E} \sum_{t^k \in (t-dt,t\,)} \delta(t - t_i^k)$$

spike count

$$N_{spikes}^{I\ in\ n}(t) = \sum_{i=1}^{N_I} \sum_{t^k \in (t-dt,t\,)} \delta(t - t_i^k)$$

Spikes $\delta(t - t_i^k)$

Spikes $\delta(t - t_i^k)$

NEST Spiking Network node n

Excitatory Population E

Inhibitory Population I

Human Brain Project

Modifications to the TVB simulator

- TVB-SpikeNet interface properties and configurations
- Dynamical non-state variables (e.g. $R_{\acute{n}}^{E/I}(t)$) that need to be updated in a model specific manner after integration of each time step
- Model specific modifications (e.g. $Rin_{\acute{n}}^{E/I}(t)$ variables to smooth instantaneous spike rate coming from NEST via linear integration acting as a low pass filter)

- Integration loop
  1. TVB to SpikeNet coupling
  2. TVB integration

     SpikeNet integration
  3. (Optional) SpikeNet to TVB update
  4. Compute TVB large-scale coupling and update stimulus, if any
  5. Update non-state dynamical variables (if any)
  6. Update TVB history buffer
  7. Generate TVB monitor outputs

Human Brain Project

References

1. Ritter P, Schirner M, McIntosh AR, Jirsa VK. 2013. The Virtual Brain integrates computational modeling and multimodal neuroimaging. Brain Connectivity 3:121–145.
2. Sanz Leon P, Knock SA, Woodman MM, Domide L, Mersmann J, McIntosh AR, Jirsa V. 2013. The Virtual Brain: a simulator of primate brain network dynamics. Frontiers in Neuroinformatics 7:10.
3. Jordan, Jakob, Mørk, Håkon, Vennemo, Stine Brekke, Terhorst, Dennis, Peyser, Alexander, Ippen, Tammo, … Plesser, Hans Ekkehard. (2019, June 27). NEST 2.18.0 (Version 2.18.0). Zenodo.
4. Deco G, Ponce-Alvarez A, Mantini D, Romani GL, Hagmann P, Corbetta M. 2013. Resting-State Functional Connectivity Emerges from Structurally and Dynamically Shaped Slow Linear Fluctuations. The Journal of Neuroscience 33(27): 11239 –11252.
5. Perdikis D, Schirner M, Diaz-Cortes, M-A, Domide L, Mersmann J, Ritter P (*in prep*).

THANK YOU!

www.humanbrainproject.eu

@humanbrainproj

@humanbrainproj

HumanBrainProject

Co-funded by
the European Union