TVB-ANNarchy: Bridging multiscale activity by co-simulation

Step-by-step learn how to perform a co-simulation embedding spiking neural networks into large-scale brain networks using TVB.

Izhikevich Spiking network model in ANNarchy

For every neuron i in region node n modelled in ANNarchy as a spiking network:

Membrane

potential:

$$\dot{V}_m = n_2 V_m^2 + n_1 V_m + n_0 140 - U_m/C - g_{AMPA}(V_m - E_{AMPA}) - g_{GABA}(V_m - E_{GABA}) -$$

where the conductances follow the equations:

$$\dot{g}_{AMPA} = -g_{AMPA}/\tau_{AMPA} + \left[\sum_k \delta(t - t_k)\right]_{Exc}$$

$$\dot{g}_{GABA} = -g_{GABA}/\tau_{GABA} + \left[\sum_k \delta(t - t_k)\right]_{Inh}$$

$$\dot{g}_{BASE} = -g_{BASE}/\tau_{BASE} + \left[\sum_k \delta(t - t_k)\right]_{BASE}$$

and recovery variable:

$$\dot{U}_m = a(bV_m - U_m)$$

When $V_m > V_{th}$, V_m is set to c, and U_m is incremented by d.

WORKFLOW:

In [1]:
```
from collections import OrderedDict
import time
import numpy as np

from tvb.basic.profile import TvbProfile
TvbProfile.set_profile(TvbProfile.LIBRARY_PROFILE)

from tvb_multiscale.tvb_annarchy.config import *
home_path = "/home/docker/packages/tvb-multiscale/examples"
working_path = os.path.join(home_path, "notebooks")
data_path = os.path.join(home_path, "data")
config = Config(output_base=os.path.join(working_path, "outputs_Izhikevich_an
config.figures.SHOW_FLAG = True
config.figures.SAVE_FLAG = True
config.figures.FIG_FORMAT = 'png'
config.figures.DEFAULT_SIZE= config.figures.NOTEBOOK_SIZE
FIGSIZE = config.figures.DEFAULT_SIZE

from tvb_multiscale.core.plot.plotter import Plotter
plotter = Plotter(config.figures)
```

```
# For interactive plotting:
# %matplotlib notebook

# Otherwise:
%matplotlib inline
```

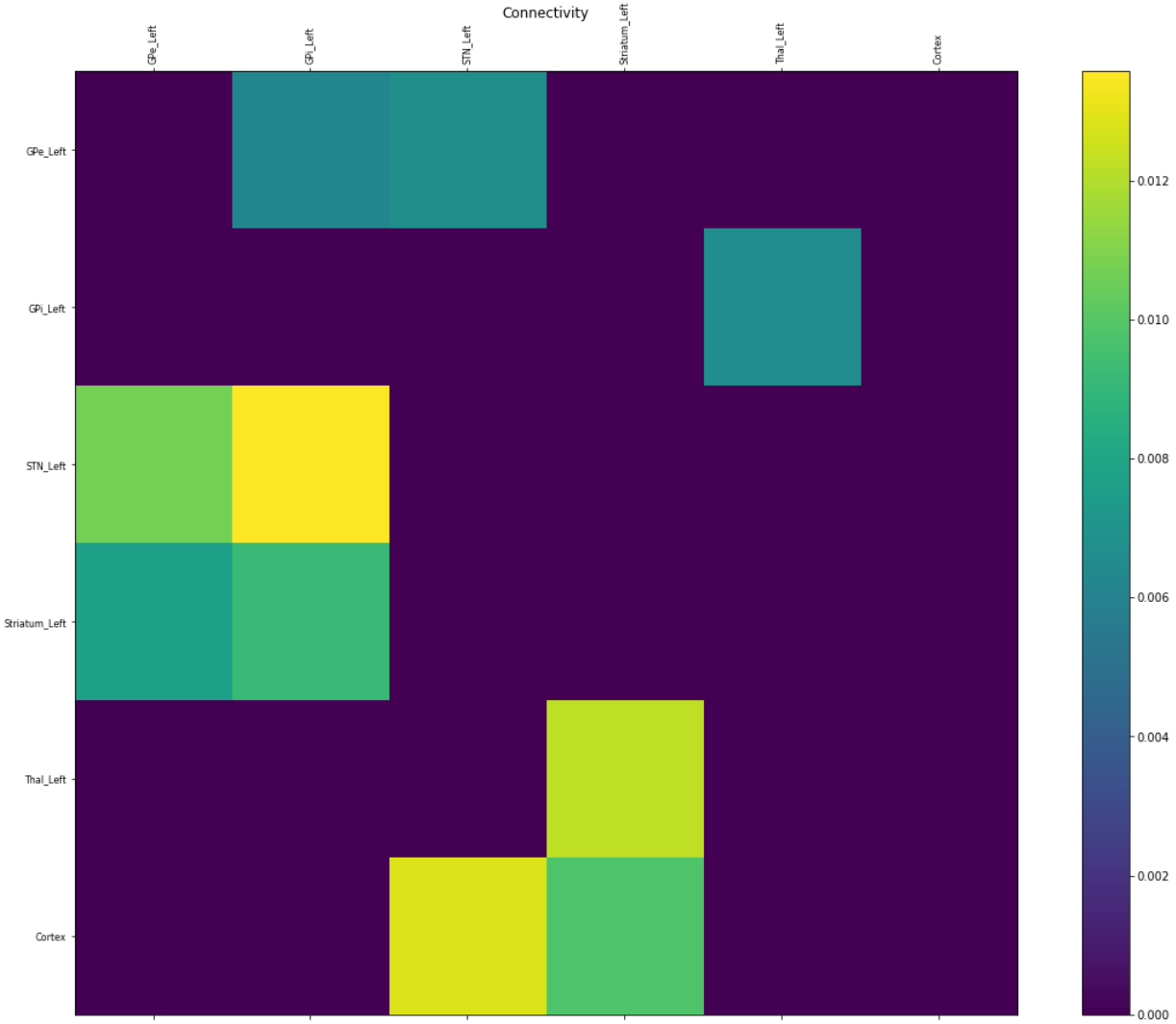
1. Load structural data (minimally a TVB connectivity) & prepare TVB simulator (region mean field model, integrator, monitors etc)

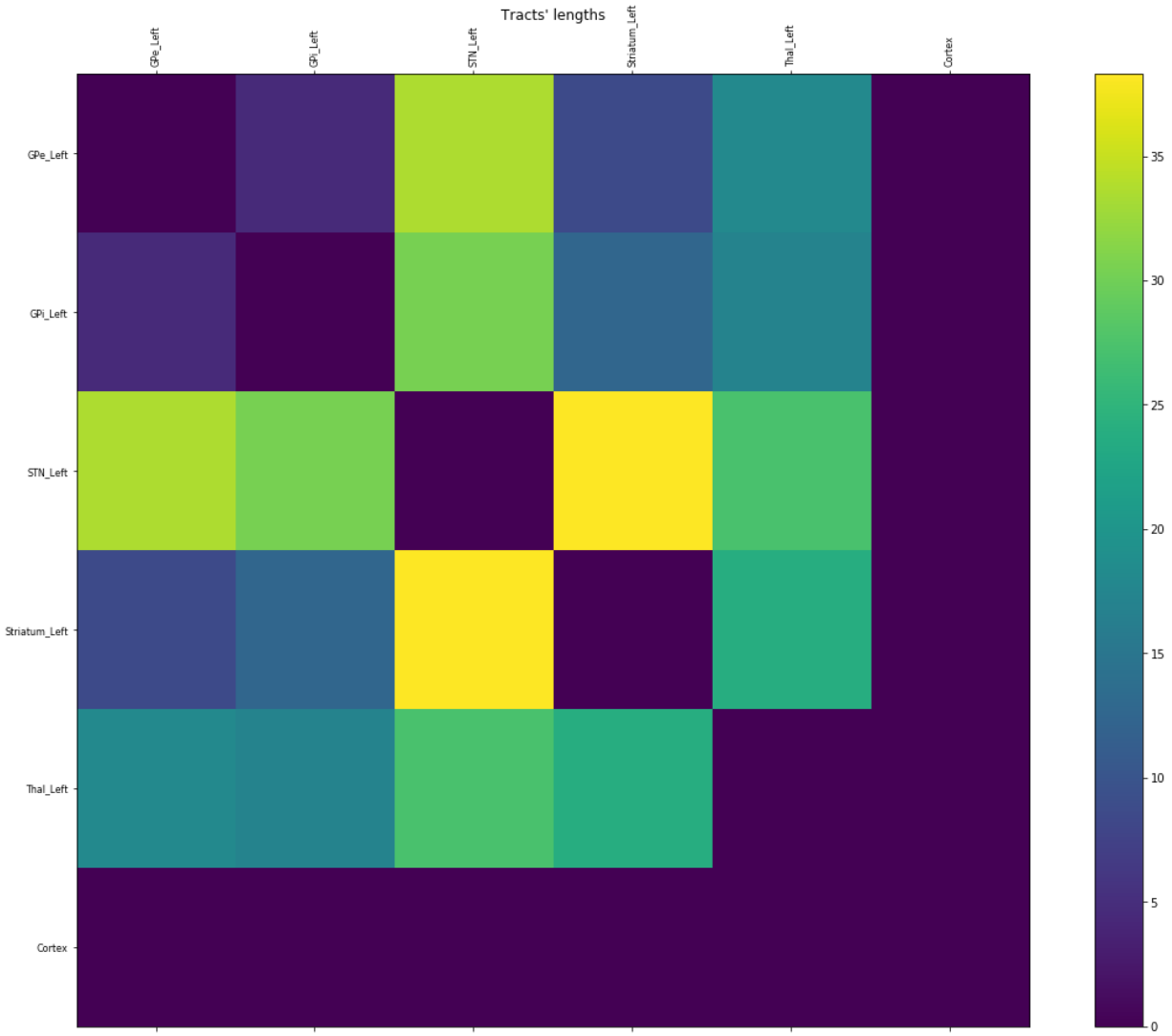
In [2]:
```
from tvb.simulator.models.reduced_wong_wang_exc_io import ReducedWongWangExcI

# -------------------------------------------------------------------------
# ----Uncomment below to modify the simulator by changing the default options
# -------------------------------------------------------------------------

from tvb.datatypes.connectivity import Connectivity
from tvb.simulator.cosimulator import CoSimulator
from tvb.simulator.integrators import HeunStochastic
from tvb.simulator.monitors import Raw  # , Bold, EEG


conn_path = os.path.join(data_path, "basal_ganglia_conn_incl_cortex")

w=np.loadtxt(os.path.join(conn_path, "opti_CON1_lh_weights_incl_cortex.txt"))
c=np.loadtxt(os.path.join(conn_path, "aal_plus_BG_centers_incl_cortex.txt"),
rl= np.loadtxt(os.path.join(conn_path, "aal_plus_BG_centers_incl_cortex.txt")
t= np.loadtxt(os.path.join(conn_path, "BGplusAAL_tract_lengths_incl_cortex.tx

# Keep only the BG and a single Cortex node:
c = c[:11]
rl = rl[:11]
rl[10] = "Cortex"
w = w[:11][:, :11]
t = t[:11][:, :11]

# Keep only left hemisphere and the Cortex:
inds = np.arange(0,10,2).astype("i").tolist() + [10]
c = c[inds]
rl = rl[inds]
print("Region labels:\n%s" % rl)
# 0. GPe_Left, 1. GPi_Left, 2. STN_Left, 3. Striatum_Left, 4. Thal_Left, 5. C
w = w[inds][:, inds]
t = t[inds][:, inds]

#load the optimized weights to use for iSN and Cortex connections
import scipy.io as sio
weights=sio.loadmat(os.path.join(conn_path,"OutputSim_Patient01.mat")) # weig

# % loadedParams ={
# %               'D1GPi_probs': probs[0],
# %               'D1GPi_weights'  : weights[0],
# %               'D2GPe_probs'   : probs[1],
# %               'D2GPe_weights'  : weights[1],
# %               'GPeSTN_probs'   : probs[2],
# %               'GPeSTN_weights'  : weights[2],
# %               'STNGPe_probs'   : probs[3],
# %               'STNGPe_weights'  : weights[3],
# %               'STNGPi_probs'   : probs[4],
# %               'STNGPi_weights' : weights[4],
```

```
# %              'GPeGPi_probs'   : probs[5],
# %              'GPeGPi_weights'  : weights[5],
# %              'GPeGPe_probs'   : probs[6],
# %              'GPeGPe_weights'  : weights[6],
# %              'GPiGPi_probs'   : probs[7],
# %              'GPiGPi_weights'  : weights[7],
# %              'GPiThal_probs'   : probs[8],
# %              'GPiThal_weights'  : weights[8],
# %              'ThaliSN_probs'   : probs[9],
# %              'ThaliSN_weights'  : weights[9],
# %              'ThaldSN_probs'   : probs[10],
# %              'ThaldSN_weights'  : weights[10],
# %              'dSNdSN_probs'   : probs[11],
# %              'dSNdSN_weights'  : weights[11],
# %              'iSNiSN_probs'   : probs[12],
# %              'iSNiSN_weights'  : weights[12],
# %              'CdSN_probs'   : probs[13],
# %              'CdSN_weights'  : weights[13],
# %              'CiSN_probs'   : probs[14],
# %              'CiSN_weights'  : weights[14],
# %              'CSTN_probs'   : probs[15],
# %              'CSTN_weights'  : weights[15],
# %              'V1Inh_probs'    : probs[16],
# %              'V1Inh_weights'  : weights[16],
# %              'InhV1_probs'    : probs[17],
# %              'InhV1_weights'  : weights[17],
# %              'InhInh_probs'   : probs[18],
# %              'InhInh_weights'  : weights[18]}

# dSN = dSN, iSN = iSN from now on

wGPeGPe = weights["X"][0, 6+19]   # "GPe" -> "GPe"
wGPiGPi = weights["X"][0, 7+19]   # "GPe" -> "GPe"
wdSNdSN = weights["X"][0, 11+19]   # "IdSN" -> "IdSN"
wiSNiSN = weights["X"][0, 12+19]   # "IiSN" -> "IiSN"
wThiSN = weights["X"][0, 9+19]    # "Eth" -> "IiSN"
# wThdSNtoThiSN = wThiSN / w[4, 3]
w[5, 2] = weights["X"][0, 15+19]  # "CxE" -> "Estn"
w[5, 3] = weights["X"][0, 13+19]  # "CxE" -> "IdSN"
wCrtxiSN = weights["X"][0, 14+19] # "CxE" -> "IiSN"
# wCrtxdSNtoCrtxiSN = weights["X"][0, 14+19] / w[5, 3]
wCtxEtoI = weights["X"][0, 16+19] # "CxE" -> "CxI"
wCtxItoE = weights["X"][0, 17+19] # "CxI" -> "CxE"
wCtxItoI = weights["X"][0, 18+19] # "CxI" -> "CxI"

# Finally form the TVB Connectivity
connectivity=Connectivity(region_labels=rl, weights=w, centres=c, tract_lengt

# Normalize connectivity weights
# connectivity.weights = connectivity.scaled_weights(mode="region")
# connectivity.weights /= np.percentile(connectivity.weights, 99)
# connectivity.weights[connectivity.weights > 1.0] = 1.0
connectivity.speed = np.array([4.0])
connectivity.configure()

#white_matter_coupling = coupling.Linear(a=0.014)
# Create a TVB simulator and set all desired inputs
# (connectivity, model, surface, stimuli etc)
# We choose all defaults in this example
simulator = CoSimulator()
#simulator.use_numba = False
model_params = {}
simulator.model = ReducedWongWangExcIO(**model_params)

simulator.connectivity = connectivity
```

```
simulator.integrator = HeunStochastic()
simulator.integrator.dt = 0.1
simulator.integrator.noise.nsig = np.array([0.001])

mon_raw = Raw(period=1.0)  # ms
simulator.monitors = (mon_raw, )

plotter.plot_tvb_connectivity(simulator.connectivity);
```

```
Region labels:
['GPe_Left' 'GPi_Left' 'STN_Left' 'Striatum_Left' 'Thal_Left' 'Cortex']
```





2. Build and connect the ANNarchy network model (networks of spiking neural populations for fine-scale regions, stimulation devices, spike detectors etc)

In [3]:
```
from tvb_multiscale.tvb_annarchy.annarchy_models.builders.models.basal_gangli
from tvb_multiscale.tvb_annarchy.annarchy.models import Izhikevich_Hamker

# Select the regions for the fine scale modeling with ANNarchy spiking networ
#including cortex node:
spiking_nodes_ids = [0, 1, 2, 3, 4, 5]  # the indices of fine scale regions m

# Build a ANNarchy network model with the corresponding builder
ann_model_builder = BasalGangliaIzhikevichBuilder(simulator, spiking_nodes_id
#                                                   dt=float(simulator.integ
#                                                   weights=np.array(simulat
#                                                   delays=np.array(simulato
#                                                   region_labels=np.array(s
#                                                   model=simulator.model,
#                                                   coupling_a=float(simulat
#                                                   G=float(simulator.model.
                                                  )



# Using all default parameters for this example


# or...
```

```
# # ------------------------------------------------------------------------
# # ----Uncomment below to modify the builder by changing the default options
# # ------------------------------------------------------------------------
from copy import deepcopy

population_neuron_model = Izhikevich_Hamker

ann_model_builder.population_order = 200 # reduce for speed

# When any of the properties model, params and scale below depends on regions
# set a handle to a function with
# arguments (region_index=None) returning the corresponding property

ann_model_builder.params_common = \
    {"E_ampa": 0.0, "E_gaba": -90.0, "v_th": 30.0, "c": -65.0,
     "C": 1.0, "I": 0.0,
     "tau_syn": 1.0, "tau_ampa": 10.0, "tau_gaba": 10.0,
     "n0": 140.0, "n1": 5.0, "n2": 0.04}

ann_model_builder._paramsI = deepcopy(ann_model_builder.params_common)
ann_model_builder._paramsI.update({"a": 0.005, "b": 0.585, "d": 4.0})
ann_model_builder._paramsE = deepcopy(ann_model_builder.params_common)
ann_model_builder.paramsStr = deepcopy(ann_model_builder.params_common)
ann_model_builder.paramsStr.update({"v_th": 40.0, "C": 50.0,
                                    "n0": 61.65, "n1": 2.59, "n2": 0.02,
                                    "a": 0.05, "b": -20.0, "c": -55.0, "d": 3

ann_model_builder.Igpe_nodes_ids = [0]
ann_model_builder.Igpi_nodes_ids = [1]
ann_model_builder.Estn_nodes_ids = [2]
ann_model_builder.Eth_nodes_ids = [4]
ann_model_builder.Istr_nodes_ids = [3]
#including cortex node:
ann_model_builder.Crtx_nodes_ids = [5]

I_nodes_ids = ann_model_builder.Igpe_nodes_ids + ann_model_builder.Igpi_nodes_
E_nodes_ids = ann_model_builder.Estn_nodes_ids + ann_model_builder.Eth_nodes_

# #including cortex node: we do not need any other external stimulation
# ann_model_builder.Estn_stim = {"rate": 500.0, "weight": 0.009}
# ann_model_builder.Igpe_stim = {"rate": 100.0, "weight": 0.015}
# ann_model_builder.Igpi_stim = {"rate": 700.0, "weight": 0.02}


def paramsE_fun(node_id):
    paramsE = deepcopy(ann_model_builder._paramsE)
    if node_id in ann_model_builder.Estn_nodes_ids:
        paramsE.update({"a": 0.005, "b": 0.265, "d": 2.0, "I": 3.0})  # dicti
    elif node_id in ann_model_builder.Eth_nodes_ids:
        paramsE.update({"a": 0.02, "b": 0.25, "d": 0.05, "I": 3.5}) # diction
    elif node_id in ann_model_builder.Crtx_nodes_ids:
        paramsE.update({"a": 0.02, "b": 0.2, "d": 6.0,"c": -72.0, "I": 50.0})
    return paramsE

def paramsI_fun(node_id):
    # For the moment they are identical, unless you differentiate the noise p
    paramsI = deepcopy(ann_model_builder._paramsI)
    if node_id in ann_model_builder.Igpe_nodes_ids:
        paramsI.update({"I": 12.0})
    elif node_id in ann_model_builder.Igpi_nodes_ids:
        paramsI.update({"I": 30.0})
    elif node_id in ann_model_builder.Crtx_nodes_ids:
        paramsI.update({"c": -72.0,"a": 0.02, "b": 0.2, "d": 6.0, "I_e": 0.0}
    return paramsI
```

```
# Populations' configurations
# When any of the properties model, params and scale below depends on regions
# set a handle to a function with
# arguments (region_index=None) returning the corresponding property
ann_model_builder.populations = [
    {"label": "E", "model": population_neuron_model,
     "params":  paramsE_fun,
     "nodes": E_nodes_ids,  # Estn in [2], Eth in [4], Cortex in [5]
     "scale": lambda node_id: 3.0 if node_id in ann_model_builder.Crtx_nodes_
    {"label": "I", "model": population_neuron_model,
     "params": paramsI_fun,
     "nodes": I_nodes_ids,  # Igpe in [0], Igpi in [1], Cortex in [5]
     "scale": lambda node_id: 0.75 if node_id in ann_model_builder.Crtx_nodes_
    {"label": "IdSN", "model": population_neuron_model,
     "params": ann_model_builder.paramsStr,
     "nodes": ann_model_builder.Istr_nodes_ids,  # IdSN in [3]
     "scale": 1.0},
    {"label": "IiSN", "model": population_neuron_model,   # IiSN in [3]
     "params": ann_model_builder.paramsStr,
     "nodes": ann_model_builder.Istr_nodes_ids,  # None means "all"
     "scale": 1.0}
]

# Within region-node connections
# When any of the properties model, conn_spec, weight, delay, receptor_type b
# set a handle to a function with
# arguments (region_index=None) returning the corresponding property

synapse_model = "DefaultSpikingSynapse"
conn_spec = {'method': "all_to_all", "allow_self_connections": True, "force_m

within_node_delay = 1.0

class WeightFun(object):

    def __init__(self, wGPeGPe, wGPiGPi, wCtxItoI):
        self.wGPeGPe = np.abs(wGPeGPe)
        self.wGPiGPi = np.abs(wGPiGPi)
        self.wCtxItoI = np.abs(wCtxItoI)

    def __call__(self, node):
        if node == 0:
            return self.wGPeGPe  # GPe -> GPe
        elif node == 1:
            return self.wGPiGPi  # GPi -> GPi
        elif node == 5:
            return self.wCtxItoI  # CxI -> CxI


# for each connection, we have a different probability
ann_model_builder.populations_connections = [
     #        source   ->   target
    {"source": "I", "target": "I",  # I -> I This is a self-connection for po
     "synapse_model": synapse_model, "conn_spec": conn_spec,#.update({"p": 0.
     "weight": WeightFun(wGPeGPe, wGPiGPi, wCtxItoI), "delay": within_node_de
     "receptor_type": "gaba", "nodes": I_nodes_ids},  # None means apply to a
    {"source": "IdSN", "target": "IdSN",  # IdSN -> IdSN This is a self-conne
     "synapse_model": synapse_model, "conn_spec": conn_spec,
     "weight": wdSNdSN, "delay": within_node_delay,
     "receptor_type": "gaba", "nodes": ann_model_builder.Istr_nodes_ids},
    {"source": "IiSN", "target": "IiSN",  # IiSN -> IiSN This is a self-conne
     "synapse_model": synapse_model, "conn_spec": conn_spec,
     "weight": wiSNiSN, "delay": within_node_delay,
     "receptor_type": "gaba", "nodes": ann_model_builder.Istr_nodes_ids},
```

```
    {"source": "E", "target": "I",          # "CxE" -> "CxI" #
     "synapse_model": synapse_model, "conn_spec": conn_spec,
     "weight": wCtxEtoI, "delay": within_node_delay,
     "receptor_type": "ampa", "nodes": ann_model_builder.Crtx_nodes_ids},  # 
    {"source": "I", "target": "E",          # "CxI" -> "CxE"
     "synapse_model": synapse_model, "conn_spec": conn_spec,
     "weight": wCtxItoE, "delay": within_node_delay,
     "receptor_type": "gaba", "nodes": ann_model_builder.Crtx_nodes_ids}  # N
    ]



# Among/Between region-node connections
# Given that only the AMPA population of one region-node couples to
# all populations of another region-node,
# we need only one connection type

# When any of the properties model, conn_spec, weight, delay, receptor_type b
# depends on regions, set a handle to a function with
# arguments (source_region_index=None, target_region_index=None)

from tvb_multiscale.core.spiking_models.builders.templates import scale_tvb_w

# NOTE!!! TAKE CARE OF DEFAULT simulator.coupling.a!
ann_model_builder.global_coupling_scaling = 1.0 # ann_model_builder.coupling_
# if we use Reduced Wong Wang model, we also need to multiply with the global
# ann_model_builder.global_coupling_scaling *= ann_model_builder.G

class TVBWeightFun(object):
    tvb_weights = np.array([])
    global_coupling_scaling = 1.0

    def __init__(self, tvb_weights, global_coupling_scaling=1.0):
        self.tvb_weights = tvb_weights
        self.global_coupling_scaling = global_coupling_scaling

    def __call__(self, source_node, target_node):
        return scale_tvb_weight(source_node, target_node, self.tvb_weights,
                                scale=self.global_coupling_scaling)

tvb_delay_fun = \
    lambda source_node, target_node: \
        np.maximum(ann_model_builder.tvb_dt, tvb_delay(source_node, target_no

# Total excitatory spikes of one region node will be distributed to
ann_model_builder.nodes_connections = [
    #        source    ->     target
    {"source": "IdSN", "target": "I",             # "IdSN" -> "Igpi"
     "synapse_model": synapse_model, "conn_spec": conn_spec,
     "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
     "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
     "receptor_type": "gaba",
     "source_nodes": ann_model_builder.Istr_nodes_ids,
     "target_nodes": ann_model_builder.Igpi_nodes_ids},  # None means apply t
    {"source": "IiSN", "target": "I",            # "IiSN" -> "Igpe"
     "synapse_model": synapse_model, "conn_spec": conn_spec,
     "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
     "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
     "receptor_type": "gaba",
     "source_nodes": ann_model_builder.Istr_nodes_ids,
     "target_nodes": ann_model_builder.Igpe_nodes_ids},  # None means apply t
    {"source": "I", "target": "I",             # "Igpe" -> "Igpi"
     "synapse_model": synapse_model, "conn_spec": conn_spec,
     "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
     "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
```

```
        "receptor_type": "gaba",
        "source_nodes": ann_model_builder.Igpe_nodes_ids,
        "target_nodes": ann_model_builder.Igpi_nodes_ids},  # None means apply t
       {"source": "I", "target": "E",              # "Igpi" -> "Eth"
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "gaba",
        "source_nodes": ann_model_builder.Igpi_nodes_ids,
        "target_nodes": ann_model_builder.Eth_nodes_ids},  # None means apply to
       {"source": "I", "target": "E",             # "Igpe" -> "Estn"
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "gaba",
        "source_nodes": ann_model_builder.Igpe_nodes_ids,
        "target_nodes": ann_model_builder.Estn_nodes_ids},  # None means apply t
       {"source": "E", "target": "IdSN",   # "Eth" -> ["IdSN"]
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "ampa",
        "source_nodes": ann_model_builder.Eth_nodes_ids,
        "target_nodes": ann_model_builder.Istr_nodes_ids},  # None means apply t
       {"source": "E", "target": "IiSN",   # "Eth" -> ["IiSN"]
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": wThiSN,
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "ampa",
        "source_nodes": ann_model_builder.Eth_nodes_ids,
        "target_nodes": ann_model_builder.Istr_nodes_ids},  # No
       {"source": "E", "target": "I",          # "Estn" -> ["Igpe", "Igpi"]
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "ampa",
        "source_nodes": ann_model_builder.Estn_nodes_ids,
        "target_nodes": ann_model_builder.Igpe_nodes_ids + ann_model_builder.Igp
#       {"source": "E", "target": "E",          # "CxE" -> "Eth"
#        "model": synapse_model, "conn_spec": conn_spec,
#        "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builde
#        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, t
#        "receptor_type": 0,
#        "source_nodes": ann_model_builder.Crtx_nodes_ids,
#        "target_nodes": ann_model_builder.Eth_nodes_ids},  # None means apply
       {"source": "E", "target": "E",          # "CxE" -> "Estn"
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "ampa",
        "source_nodes": ann_model_builder.Crtx_nodes_ids,
        "target_nodes": ann_model_builder.Estn_nodes_ids},  # None means apply t
       {"source": "E", "target": "IdSN",          # "CxE" -> "IdSN"
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": TVBWeightFun(ann_model_builder.tvb_weights, ann_model_builder.
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "ampa",
        "source_nodes": ann_model_builder.Crtx_nodes_ids,
        "target_nodes": ann_model_builder.Istr_nodes_ids},  # None means apply t
       {"source": "E", "target": "IiSN",          # "CxE" -> "IiSN"
        "synapse_model": synapse_model, "conn_spec": conn_spec,
        "weight": wCrtxiSN, # TVBWeightFun(ann_model_builder.tvb_weights,
                            # wCrtxdSNtoCrtxiSN * ann_model_builder.global_co
        "delay": lambda source_node, target_node: tvb_delay_fun(source_node, tar
        "receptor_type": "ampa",
```

```
        "source_nodes": ann_model_builder.Crtx_nodes_ids,
        "target_nodes": ann_model_builder.Istr_nodes_ids}  # None means apply to
        ]

# Creating  devices to be able to observe ANNarchy activity:

ann_model_builder.output_devices = []

period = 1.0

# Creating  devices to be able to observe ANNarchy activity:
params = ann_model_builder.config.ANNARCHY_OUTPUT_DEVICES_PARAMS_DEF["SpikeMo
params["period"] = period
for pop in ann_model_builder.populations:
    connections = OrderedDict({})
    #                       label <- target population
    params["label"] = pop["label"] + "_spikes"
    connections[params["label"]] = pop["label"]
    ann_model_builder.output_devices.append(
        {"model": "SpikeMonitor", "params": deepcopy(params),
         "connections": connections, "nodes": pop["nodes"]})  # None means ap

# Labels have to be different for every connection to every distinct populati
# params for baladron implementation commented out for the moment
# TODO: use baladron neurons
params = ann_model_builder.config.ANNARCHY_OUTPUT_DEVICES_PARAMS_DEF["Monitor
params.update({"period": period,  'record_from': ["v", "u", "I_syn", "I_syn_e
for pop in ann_model_builder.populations:
    connections = OrderedDict({})
    #                label    <- target population
    connections[pop["label"]] = pop["label"]
    params["label"] = pop["label"]
    ann_model_builder.output_devices.append(
        {"model": "Monitor", "params": deepcopy(params),
         "connections": connections, "nodes": pop["nodes"]})  # None means ap


#Create a spike stimulus input device
ann_model_builder.input_devices = [
#             {"model": "PoissonPopulation",
#              "params": {"rates": self.Estn_stim["rate"], "geometry": popula
#              "connections": {"BaselineEstn": ["E"]},  # "Estn"
#              "nodes": self.Estn_nodes_ids,  # None means apply to all
#              "weights": self.Estn_stim["weight"], "delays": 0.0, "receptor_
#             {"model": "PoissonPopulation",
#              "params": {"rates": self.Igpe_stim["rate"], "geometry": popula
#              "connections": {"BaselineIgpe": ["I"]},  # "Igpe"
#              "nodes": self.Igpe_nodes_ids,  # None means apply to all
#              "weights": self.Igpe_stim["weight"], "delays": 0.0, "receptor_
#             {"model": "PoissonPopulation",
#              "params": {"rates": self.Igpi_stim["rate"], "geometry": popula
#              "connections": {"BaselineIgpi": ["I"]},  # "Igpi"
#              "nodes": self.Igpi_nodes_ids,  # None means apply to all
#              "weights": self.Igpi_stim["weight"], "delays": 0.0, "receptor_
            # {"model": "ACCurrentInjector",
            #  "params": {"frequency": 30.0, "phase": 0.0, "amplitude": 1.0,
            #  "connections": {"DBS_Estn": ["E"]},  # "Estn"
            #  "nodes": self.Estn_nodes_ids,  # None means apply to all
            #  "weights": 1.0, "delays": 0.0}
       ]  #

# -----------------------------------------------------------------------
# -----------------------------------------------------------------------
# -----------------------------------------------------------------------
```

```
ann_network = ann_model_builder.build_spiking_network()
```

```
ANNarchy 4.6 (4.6.9.7) on linux (posix).
2020-11-26 15:05:42,080 - INFO - tvb_multiscale.tvb_annarchy.annarchy_models.b
uilders.base - Loading an ANNarchy instance...
2020-11-26 15:05:42,080 - INFO - tvb_multiscale.tvb_annarchy.annarchy_models.b
uilders.base - Loading an ANNarchy instance...
2020-11-26 15:05:42,087 - INFO - tvb_multiscale.tvb_annarchy.annarchy_models.b
uilders.base - Cleaning ANNarchy compilation directory, if any...
2020-11-26 15:05:42,087 - INFO - tvb_multiscale.tvb_annarchy.annarchy_models.b
uilders.base - Cleaning ANNarchy compilation directory, if any...
2020-11-26 15:05:44,638 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for E_spikes created!
2020-11-26 15:05:44,638 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for E_spikes created!
2020-11-26 15:05:44,656 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for I_spikes created!
2020-11-26 15:05:44,656 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for I_spikes created!
2020-11-26 15:05:44,673 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for IdSN_spikes created!
2020-11-26 15:05:44,673 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for IdSN_spikes created!
2020-11-26 15:05:44,689 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for IiSN_spikes created!
2020-11-26 15:05:44,689 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model SpikeM
onitor for IiSN_spikes created!
2020-11-26 15:05:44,708 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for E created!
2020-11-26 15:05:44,708 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for E created!
2020-11-26 15:05:44,736 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for I created!
2020-11-26 15:05:44,736 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for I created!
2020-11-26 15:05:44,756 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for IdSN created!
2020-11-26 15:05:44,756 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for IdSN created!
2020-11-26 15:05:44,772 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for IiSN created!
2020-11-26 15:05:44,772 - INFO - tvb_multiscale.core.spiking_models.devices -
<class 'tvb_multiscale.core.spiking_models.devices.DeviceSet'> of model Monito
r for IiSN created!
2020-11-26 15:05:44,784 - INFO - tvb_multiscale.core.spiking_models.network -
<class 'tvb_multiscale.tvb_annarchy.annarchy_models.network.ANNarchyNetwork'>
created!
2020-11-26 15:05:44,784 - INFO - tvb_multiscale.core.spiking_models.network -
<class 'tvb_multiscale.tvb_annarchy.annarchy_models.network.ANNarchyNetwork'>
created!
```

3. Configure simulator, simulate, gather results

In [4]:
```
simulation_length=110.0
transient = 10.0 # simulation_length/11
t = time.time()
ann_network.configure()  # ann_network.annarchy_instance.compile()
print("\nCompiled in %g secs!" % (time.time() - t))
print(ann_network.print_str(connectivity=False))
```

```
WARNING: Can not find python-config in the same directory as python, trying wi
th the default path...
Compiling...
OK

Compiled in 142.732 secs!

--------------------------------------------------------------------------------
----------------------

--------------------------------------------------------------------------------
----------------------

--------------------------------------------------------------------------------
----------------------
ANNarchyNetwork:

--------------------------------------------------------------------------------
----------------------

SpikingBrain - Regions: ['GPe_Left', 'GPi_Left', 'STN_Left', 'Striatum_Left',
'Thal_Left', 'Cortex']
Regions' nodes:


--------------------------------------------------------------------------------
----------------------
ANNarchyRegionNode - Label: GPe_Left
Populations ['I']:

--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: I
model: Spiking neuron
200 neurons in population with index: 0
parameters: {'a': array([0.005]), 'b': array([0.585]), 'c': array([-65.]),
'd': array([4.]), 'n0': array([140.]), 'n1': array([5.]), 'n2': array([0.04]),
'I': array([12.]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_
gaba': array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_sy
n': array([1.]), 'C': array([1.]), 'v_th': array([30.]), 'I_syn_ex': array
([0.]), 'I_syn_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]),
'g_ampa': array([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array
([-14.4]), 'r': array([0.])},

--------------------------------------------------------------------------------
----------------------
ANNarchyRegionNode - Label: GPi_Left
Populations ['I']:

--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: I
model: Spiking neuron
200 neurons in population with index: 1
parameters: {'a': array([0.005]), 'b': array([0.585]), 'c': array([-65.]),
'd': array([4.]), 'n0': array([140.]), 'n1': array([5.]), 'n2': array([0.04]),
'I': array([30.]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_
gaba': array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_sy
n': array([1.]), 'C': array([1.]), 'v_th': array([30.]), 'I_syn_ex': array
([0.]), 'I_syn_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]),
```

```
'g_ampa': array([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array
([-14.4]), 'r': array([0.])},

--------------------------------------------------------------------------------
----------------------
ANNarchyRegionNode - Label: STN_Left
Populations ['E']:

--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: E
model: Spiking neuron
200 neurons in population with index: 2
parameters: {'a': array([0.005]), 'b': array([0.265]), 'c': array([-65.]),
'd': array([2.]), 'n0': array([140.]), 'n1': array([5.]), 'n2': array([0.04]),
'I': array([3.]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_g
aba': array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_syn':
array([1.]), 'C': array([1.]), 'v_th': array([30.]), 'I_syn_ex': array([0.]),
'I_syn_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]), 'g_amp
a': array([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array([-14.
4]), 'r': array([0.])},

--------------------------------------------------------------------------------
----------------------
ANNarchyRegionNode - Label: Striatum_Left
Populations ['IdSN', 'IiSN']:

--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: IdSN
model: Spiking neuron
200 neurons in population with index: 3
parameters: {'a': array([0.05]), 'b': array([-20.]), 'c': array([-55.]), 'd':
array([377.]), 'n0': array([61.65]), 'n1': array([2.59]), 'n2': array([0.02]),
'I': array([0.]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_g
aba': array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_syn':
array([1.]), 'C': array([50.]), 'v_th': array([40.]), 'I_syn_ex': array([0.]),
'I_syn_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]), 'g_amp
a': array([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array([-14.
4]), 'r': array([0.])},
--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: IiSN
model: Spiking neuron
200 neurons in population with index: 4
parameters: {'a': array([0.05]), 'b': array([-20.]), 'c': array([-55.]), 'd':
array([377.]), 'n0': array([61.65]), 'n1': array([2.59]), 'n2': array([0.02]),
'I': array([0.]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_g
aba': array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_syn':
array([1.]), 'C': array([50.]), 'v_th': array([40.]), 'I_syn_ex': array([0.]),
'I_syn_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]), 'g_amp
a': array([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array([-14.
4]), 'r': array([0.])},

--------------------------------------------------------------------------------
----------------------
ANNarchyRegionNode - Label: Thal_Left
Populations ['E']:

--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: E
model: Spiking neuron
200 neurons in population with index: 5
parameters: {'a': array([0.02]), 'b': array([0.25]), 'c': array([-65.]), 'd':
```

```
array([0.05]), 'n0': array([140.]), 'n1': array([5.]), 'n2': array([0.04]),
'I': array([3.5]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_
gaba': array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_sy
n': array([1.]), 'C': array([1.]), 'v_th': array([30.]), 'I_syn_ex': array
([0.]), 'I_syn_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]),
'g_ampa': array([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array
([-14.4]), 'r': array([0.])},

--------------------------------------------------------------------------------
----------------------
ANNarchyRegionNode - Label: Cortex
Populations ['E', 'I']:

--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: E
model: Spiking neuron
600 neurons in population with index: 6
parameters: {'a': array([0.02]), 'b': array([0.2]), 'c': array([-72.]), 'd': a
rray([6.]), 'n0': array([140.]), 'n1': array([5.]), 'n2': array([0.04]), 'I':
array([50.]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_gab
a': array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_syn': a
rray([1.]), 'C': array([1.]), 'v_th': array([30.]), 'I_syn_ex': array([0.]),
'I_syn_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]), 'g_amp
a': array([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array([-14.
4]), 'r': array([0.])},
--------------------------------------------------------------------------------
----------------------

ANNarchyPopulation - Label: I
model: Spiking neuron
150 neurons in population with index: 7
parameters: {'a': array([0.02]), 'b': array([0.2]), 'c': array([-72.]), 'd': a
rray([6.]), 'n0': array([140.]), 'n1': array([5.]), 'n2': array([0.04]), 'I':
array([0.]), 'tau_refrac': array([10.]), 'tau_ampa': array([10.]), 'tau_gaba':
array([10.]), 'E_ampa': array([0.]), 'E_gaba': array([-90.]), 'tau_syn': array
([1.]), 'C': array([1.]), 'v_th': array([30.]), 'I_syn_ex': array([0.]), 'I_sy
n_in': array([0.]), 'I_syn': array([0.]), 'g_base': array([0.]), 'g_ampa': arr
ay([0.]), 'g_gaba': array([0.]), 'v': array([-72.]), 'u': array([-14.4]), 'r':
array([0.])},
--------------------------------------------------------------------------------
----------------------

--------------------------------------------------------------------------------
----------------------


Input Devices:

--------------------------------------------------------------------------------
----------------------

--------------------------------------------------------------------------------
----------------------


Output Devices:

--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: E_spikes, Model: SpikeMonitor,
Devices:
--------------------------------------------------------------------------------
----------------------

E_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
```

```
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340eada20>>]}
--------------------------------------------------------------------------------
----------------------

E_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340ead940>>]}
--------------------------------------------------------------------------------
----------------------

E_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340eadb38>>]}
--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: I_spikes, Model: SpikeMonitor,
Devices:
--------------------------------------------------------------------------------
----------------------

I_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340ead8d0>>]}
--------------------------------------------------------------------------------
----------------------

I_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340eadc50>>]}
--------------------------------------------------------------------------------
----------------------

I_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340eadd68>>]}
--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: IdSN_spikes, Model: SpikeMonitor,
Devices:
--------------------------------------------------------------------------------
----------------------

IdSN_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340eaddd8>>]}
--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: IiSN_spikes, Model: SpikeMonitor,
Devices:
--------------------------------------------------------------------------------
----------------------
```

```
IiSN_spikes: ANNarchySpikeMonitor - Model: SpikeMonitor
None
parameters: {'variables': [['spike']], 'period': [1.0], 'period_offset': [0.
0], 'start': [<bound method Monitor.start of <ANNarchy.core.Monitor.Monitor ob
ject at 0x7fb340eadc18>>]}
--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: E, Model: Monitor,
Devices:
--------------------------------------------------------------------------------
----------------------

E: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c198>>]}
--------------------------------------------------------------------------------
----------------------

E: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c2b0>>]}
--------------------------------------------------------------------------------
----------------------

E: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c390>>]}
--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: I, Model: Monitor,
Devices:
--------------------------------------------------------------------------------
----------------------

I: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c400>>]}
--------------------------------------------------------------------------------
----------------------

I: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c4a8>>]}
--------------------------------------------------------------------------------
----------------------

I: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c668>>]}
```

```
--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: IdSN, Model: Monitor,
Devices:
--------------------------------------------------------------------------------
----------------------

IdSN: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c518>>]}
--------------------------------------------------------------------------------
----------------------

DeviceSet - Name: IiSN, Model: Monitor,
Devices:
--------------------------------------------------------------------------------
----------------------

IiSN: ANNarchyMonitor - Model: Monitor
None
parameters: {'variables': [['v', 'u', 'I_syn', 'I_syn_ex', 'I_syn_in', 'g_amp
a', 'g_gaba', 'g_base']], 'period': [1.0], 'period_offset': [0.0], 'start': [<
bound method Monitor.start of <ANNarchy.core.Monitor.Monitor object at 0x7fb34
0e5c550>>]}
```

In [5]:
```
ann_network.Run(simulation_length)  # ann_network.annarchy_instance.simulate(
```

```
Simulating 0.11 seconds of the network took 0.20657849311828613 seconds.

Simulated in 0.20928 secs!
```

4. Plot results and write them to HDF5 files

In [6]:
```
# set to False for faster plotting of only mean field variables and dates, ap
plot_per_neuron = False
MAX_VARS_IN_COLS = 3
MAX_REGIONS_IN_ROWS = 10
MIN_REGIONS_FOR_RASTER_PLOT = 9
# from examples.plot_write_results import plot_write_results
# populations = []
# populations_sizes = []
# for pop in ann_model_builder.populations:
#     populations.append(pop["label"])
#     populations_sizes.append(int(np.round(pop["scale"] * ann_model_builder.
# plot_write_results(results, simulator, populations=populations, populations
#                    transient=transient, tvb_state_variable_type_label="Stat
#                    tvb_state_variables_labels=simulator.model.variables_of_
#                    plot_per_neuron=plot_per_neuron, plotter=plotter, config

# If you want to see what the function above does, take the steps, one by one
try:
    # We need framework_tvb for writing and reading from HDF5 files
    from tvb_multiscale.core.io.h5_writer import H5Writer
    from tvb.contrib.scripts.datatypes.time_series import TimeSeriesRegion
    writer = H5Writer()
except:
    writer = False
```

Spiking Network plots

In [7]:
```
from tvb_multiscale.tvb_elephant.spiking_network_analyser import SpikingNetwo
```

```
# Create a SpikingNetworkAnalyzer:
spikeNet_analyzer = \
    SpikingNetworkAnalyser(spikeNet=ann_network,
                           start_time=period, end_time=simulation_length,
                           period=period, transient=transient,
                           time_series_output_type="TVB", return_data=True,
                           force_homogeneous_results=True, connectivity=simul
```

Plot spikes' raster and mean spike rates and correlations

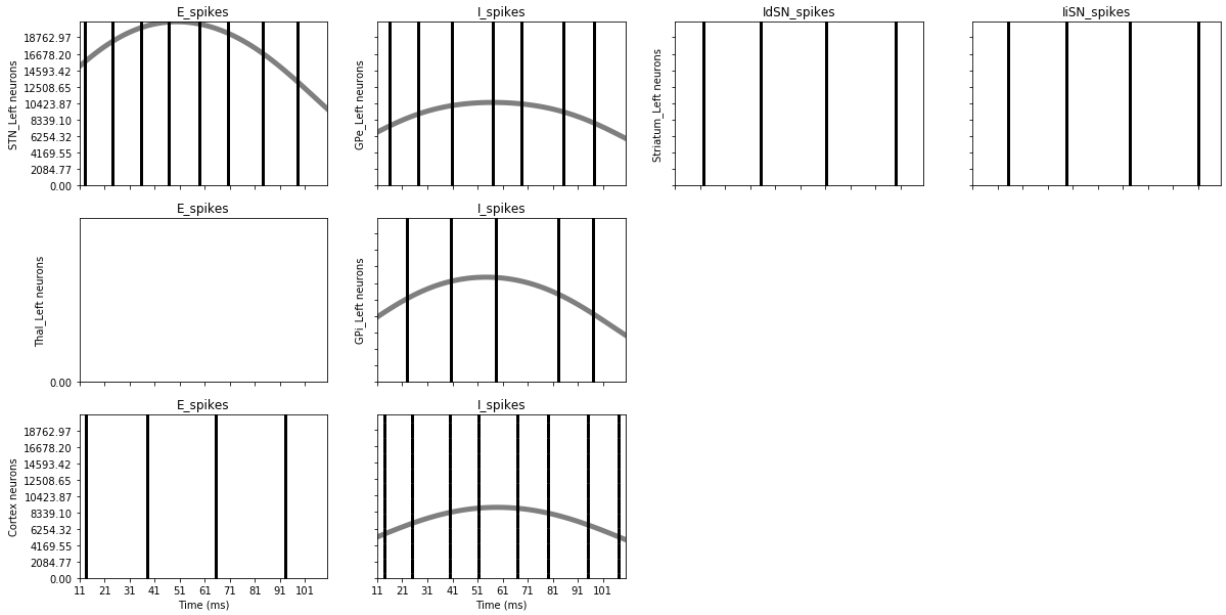
In [8]:
```
# Spikes rates and correlations per Population and Region
spikes_res = \
    spikeNet_analyzer.\
        compute_spikeNet_spikes_rates_and_correlations(
            populations_devices=None, regions=None,
            rates_methods=[], rates_kwargs=[{}],rate_results_names=[],
            corrs_methods=[], corrs_kwargs=[{}], corrs_results_names=[], bin_
            data_method=spikeNet_analyzer.get_spikes_from_device, data_kwargs
            return_devices=False
        );
```

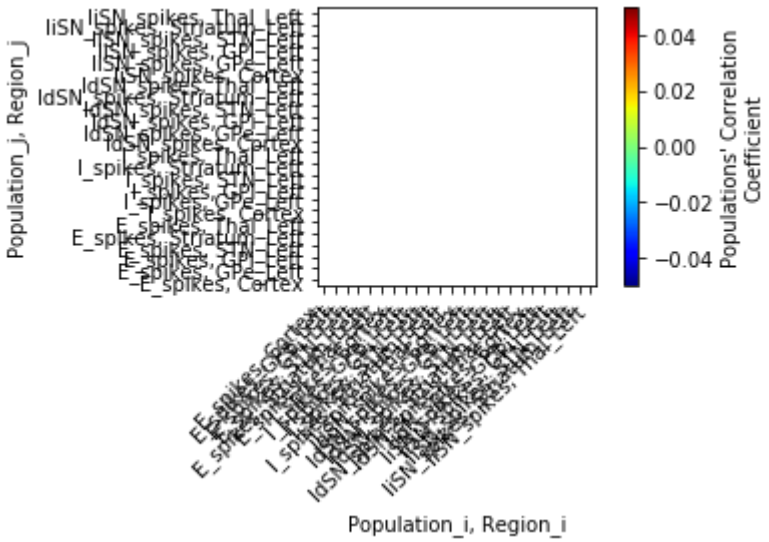
```
/home/docker/env/neurosci/lib/python3.7/site-packages/elephant/conversion.py:4
95: UserWarning: Binning discarded 10950 last spike(s) in the input spiketrai
n.
  n=n_spikes - n_spikes_binned))
/home/docker/env/neurosci/lib/python3.7/site-packages/elephant/spike_train_cor
relation.py:510: UserWarning: Detected empty spike trains (rows) in the binned
_spiketrain.
  'Detected empty spike trains (rows) in the binned_spiketrain.')
/home/docker/env/neurosci/lib/python3.7/site-packages/elephant/spike_train_cor
relation.py:517: RuntimeWarning: invalid value encountered in true_divide
  res /= (stdx.T * stdx)
```

In [9]:
```
if spikes_res:
    print(spikes_res["mean_rate"])
    print(spikes_res["spikes_correlation_coefficient"])
    # Plot spikes' rasters together with mean population's spikes' rates' tim
    if plotter:
        plotter.plot_spike_events(spikes_res["spikes"], rates=spikes_res["mea
        from tvb_multiscale.core.plot.correlations_plot import plot_correlati
        plot_correlations(spikes_res["spikes_correlation_coefficient"], plott
```

```
<xarray.DataArray "Mean Populations' Spikes' Rates" (Population: 4, Region: 6)
>
array([[24.24242424,         nan,         nan, 16.16161616,         nan,
         0.        ],
       [12.12121212, 14.14141414, 10.1010101 ,         nan,         nan,
                nan],
       [        nan,         nan,         nan,         nan,  8.08080808,
                nan],
       [        nan,         nan,         nan,         nan,  8.08080808,
                nan]])
Coordinates:
  * Region      (Region) object 'Cortex' 'GPe_Left' ... 'Thal_Left'
  * Population  (Population) object 'E_spikes' 'I_spikes' ... 'IiSN_spikes'
<xarray.DataArray "Populations' Correlation Coefficient" (Population_i: 4, Pop
ulation_j: 4, Region_i: 6, Region_j: 6)>
array([[[[nan, nan, nan, nan, nan, nan],
         [nan, nan, nan, nan, nan, nan],
         [nan, nan, nan, nan, nan, nan],
         [nan, nan, nan, nan, nan, nan],
         [nan, nan, nan, nan, nan, nan],
         [nan, nan, nan, nan, nan, nan]],

        [[nan, nan, nan, nan, nan, nan],
         [nan, nan, nan, nan, nan, nan],
         [nan, nan, nan, nan, nan, nan],
```

```
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan]],

         [[nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan]],
...
         [[nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan]],

         [[nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan]],

         [[nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan],
          [nan, nan, nan, nan, nan, nan]]]])
Coordinates:
  * Population_i  (Population_i) object 'E_spikes' 'I_spikes' ... 'IiSN_spike
s'
  * Region_i      (Region_i) object 'Cortex' 'GPe_Left' ... 'Thal_Left'
  * Population_j  (Population_j) object 'E_spikes' 'I_spikes' ... 'IiSN_spike
s'
  * Region_j      (Region_j) object 'Cortex' 'GPe_Left' ... 'Thal_Left'
2020-11-26 15:08:12,895 - ERROR - tvb.contrib.scripts.datatypes.time_series_xa
rray - Cannot access index 3 of labels ordering: ('Time', 'Population', 'Regio
n')!
2020-11-26 15:08:12,903 - ERROR - tvb.contrib.scripts.datatypes.time_series_xa
rray - Cannot access index 3 of labels ordering: ('Time', 'Population', 'Regio
n')!
```





In [10]:
```
if spikes_res and writer:
    writer.write_object(spikes_res["spikes"].to_dict(),
                        path=os.path.join(config.out.FOLDER_RES,  "Spikes") +
    writer.write_object(spikes_res["mean_rate"].to_dict(),
                        path=os.path.join(config.out.FOLDER_RES,
                                          spikes_res["mean_rate"].name) + ".h
    writer.write_tvb_to_h5(TimeSeriesRegion().from_xarray_DataArray(
                               spikes_res["mean_rate_time_series"]._data,
                                connectivity=spikes_res["mean_rate_time_series
                           os.path.join(config.out.FOLDER_RES,
                                        spikes_res["mean_rate_time_series"].t
                           recursive=False);
    writer.write_object(spikes_res["spikes_correlation_coefficient"].to_dict(
                        path=os.path.join(config.out.FOLDER_RES,
                                          spikes_res["spikes_correlation_coef
```

```
2020-11-26 15:08:16,287 - INFO - tvb_multiscale.core.io.h5_writer - Starting t
o write dict to: /home/docker/packages/tvb-multiscale/examples/notebooks/outpu
ts_Izhikevich_annarchy/res/Spikes.h5
2020-11-26 15:08:16,287 - INFO - tvb_multiscale.core.io.h5_writer - Starting t
o write dict to: /home/docker/packages/tvb-multiscale/examples/notebooks/outpu
ts_Izhikevich_annarchy/res/Spikes.h5
2020-11-26 15:08:16,328 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,328 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,337 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,337 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,350 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/STN_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> times:
[ 1.625 13.25  23.95  ... 70.125 84.15  97.825] !

2020-11-26 15:08:16,350 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/STN_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> times:
[ 1.625 13.25  23.95  ... 70.125 84.15  97.825] !

2020-11-26 15:08:16,359 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/STN_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> senders:
[  0   0   0 ... 199 199 199] !
```

```
2020-11-26 15:08:16,359 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/STN_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> senders:
[  0   0   0 ... 199 199 199] !

2020-11-26 15:08:16,369 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,369 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,379 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,379 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,387 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Thal_Left" (2 members)> dataset <cla
ss 'numpy.ndarray'> times:
[] !

2020-11-26 15:08:16,387 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Thal_Left" (2 members)> dataset <cla
ss 'numpy.ndarray'> times:
[] !

2020-11-26 15:08:16,394 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Thal_Left" (2 members)> dataset <cla
ss 'numpy.ndarray'> senders:
[] !

2020-11-26 15:08:16,394 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Thal_Left" (2 members)> dataset <cla
ss 'numpy.ndarray'> senders:
[] !

2020-11-26 15:08:16,413 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,413 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,421 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,421 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,431 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> times:
[ 1.075 13.55  38.125 ... 38.125 65.225 93.05 ] !

2020-11-26 15:08:16,431 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> times:
[ 1.075 13.55  38.125 ... 38.125 65.225 93.05 ] !

2020-11-26 15:08:16,442 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> senders:
[  0   0   0 ... 599 599 599] !

2020-11-26 15:08:16,442 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/E_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> senders:
[  0   0   0 ... 599 599 599] !
```

```
2020-11-26 15:08:16,466 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,466 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,477 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,477 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,493 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPe_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> times:
[ 3.05  15.875 27.4   ... 68.6   85.475 97.325] !

2020-11-26 15:08:16,493 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPe_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> times:
[ 3.05  15.875 27.4   ... 68.6   85.475 97.325] !

2020-11-26 15:08:16,522 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPe_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> senders:
[  0   0   0 ... 199 199 199] !

2020-11-26 15:08:16,522 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPe_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> senders:
[  0   0   0 ... 199 199 199] !

2020-11-26 15:08:16,539 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,539 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,552 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,552 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,567 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPi_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> times:
[ 1.55  22.725 40.575 ... 58.325 83.225 97.075] !

2020-11-26 15:08:16,567 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPi_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> times:
[ 1.55  22.725 40.575 ... 58.325 83.225 97.075] !

2020-11-26 15:08:16,580 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPi_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> senders:
[  0   0   0 ... 199 199 199] !

2020-11-26 15:08:16,580 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/GPi_Left" (2 members)> dataset <clas
s 'numpy.ndarray'> senders:
[  0   0   0 ... 199 199 199] !

2020-11-26 15:08:16,595 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,595 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !
```

```
2020-11-26 15:08:16,602 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,602 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,616 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> times:
[  2.55    13.75    24.875 ...  79.075  94.875 107.45 ] !

2020-11-26 15:08:16,616 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> times:
[  2.55    13.75    24.875 ...  79.075  94.875 107.45 ] !

2020-11-26 15:08:16,643 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> senders:
[  0   0   0 ... 149 149 149] !

2020-11-26 15:08:16,643 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/I_spikes/Cortex" (2 members)> dataset <class
'numpy.ndarray'> senders:
[  0   0   0 ... 149 149 149] !

2020-11-26 15:08:16,657 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,657 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,663 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,663 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,717 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/IdSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> times:
[ 3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
  3.9    22.35   45.05   71.4    99.175   3.9    22.35   45.05   71.4    99.175
```

```
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
3.9    22.35  45.05  71.4   99.175 3.9    22.35  45.05  71.4   99.175
```

```
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175] !

2020-11-26 15:08:16,717 - WARNING - tvb_multiscale.core.io.h5_writer - 
Failed to write to <HDF5 group "/IdSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> times:
[ 3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
  3.9    22.35  45.05  71.4   99.175  3.9    22.35  45.05  71.4   99.175
```

```
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175
 3.9   22.35  45.05  71.4   99.175  3.9   22.35  45.05  71.4   99.175] !

2020-11-26 15:08:16,733 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/IdSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> senders:
[  0   0   0   0   0   1   1   1   1   1   2   2   2   2   2   3   3   3
   3   3   4   4   4   4   4   5   5   5   5   5   6   6   6   6   6   7
   7   7   7   7   8   8   8   8   8   9   9   9   9   9  10  10  10  10
  10  11  11  11  11  11  12  12  12  12  12  13  13  13  13  13  14  14
  14  14  14  15  15  15  15  15  16  16  16  16  16  17  17  17  17  17
  18  18  18  18  18  19  19  19  19  19  20  20  20  20  20  21  21  21
  21  21  22  22  22  22  22  23  23  23  23  23  24  24  24  24  24  25
  25  25  25  25  26  26  26  26  26  27  27  27  27  27  28  28  28  28
  28  29  29  29  29  29  30  30  30  30  30  31  31  31  31  31  32  32
  32  32  32  33  33  33  33  33  34  34  34  34  34  35  35  35  35  35
  36  36  36  36  36  37  37  37  37  37  38  38  38  38  38  39  39  39
  39  39  40  40  40  40  40  41  41  41  41  41  42  42  42  42  42  43
  43  43  43  43  44  44  44  44  44  45  45  45  45  45  46  46  46  46
  46  47  47  47  47  47  48  48  48  48  48  49  49  49  49  49  50  50
  50  50  50  51  51  51  51  51  52  52  52  52  52  53  53  53  53  53
  54  54  54  54  54  55  55  55  55  55  56  56  56  56  56  57  57  57
  57  57  58  58  58  58  58  59  59  59  59  59  60  60  60  60  60  61
  61  61  61  61  62  62  62  62  62  63  63  63  63  63  64  64  64  64
  64  65  65  65  65  65  66  66  66  66  66  67  67  67  67  67  68  68
  68  68  68  69  69  69  69  69  70  70  70  70  70  71  71  71  71  71
  72  72  72  72  72  73  73  73  73  73  74  74  74  74  74  75  75  75
  75  75  76  76  76  76  76  77  77  77  77  77  78  78  78  78  78  79
  79  79  79  79  80  80  80  80  80  81  81  81  81  81  82  82  82  82
  82  83  83  83  83  83  84  84  84  84  84  85  85  85  85  85  86  86
  86  86  86  87  87  87  87  87  88  88  88  88  88  89  89  89  89  89
  90  90  90  90  90  91  91  91  91  91  92  92  92  92  92  93  93  93
```

```
 93  93  94  94  94  94  94  95  95  95  95  95  96  96  96  96  96  97
 97  97  97  97  98  98  98  98  98  99  99  99  99  99 100 100 100 100
100 101 101 101 101 101 102 102 102 102 102 103 103 103 103 103 104 104
104 104 104 105 105 105 105 105 106 106 106 106 106 107 107 107 107 107
108 108 108 108 108 109 109 109 109 109 110 110 110 110 110 111 111 111
111 111 112 112 112 112 112 113 113 113 113 113 114 114 114 114 114 115
115 115 115 115 116 116 116 116 116 117 117 117 117 117 118 118 118 118
118 119 119 119 119 119 120 120 120 120 120 121 121 121 121 121 122 122
122 122 122 123 123 123 123 123 124 124 124 124 124 125 125 125 125 125
126 126 126 126 126 127 127 127 127 127 128 128 128 128 128 129 129 129
129 129 130 130 130 130 130 131 131 131 131 131 132 132 132 132 132 133
133 133 133 133 134 134 134 134 134 135 135 135 135 135 136 136 136 136
136 137 137 137 137 137 138 138 138 138 138 139 139 139 139 139 140 140
140 140 140 141 141 141 141 141 142 142 142 142 142 143 143 143 143 143
144 144 144 144 144 145 145 145 145 145 146 146 146 146 146 147 147 147
147 147 148 148 148 148 148 149 149 149 149 149 150 150 150 150 150 151
151 151 151 151 152 152 152 152 152 153 153 153 153 153 154 154 154 154
154 155 155 155 155 155 156 156 156 156 156 157 157 157 157 157 158 158
158 158 158 159 159 159 159 159 160 160 160 160 160 161 161 161 161 161
162 162 162 162 162 163 163 163 163 163 164 164 164 164 164 165 165 165
165 165 166 166 166 166 166 167 167 167 167 167 168 168 168 168 168 169
169 169 169 169 170 170 170 170 170 171 171 171 171 171 172 172 172 172
172 173 173 173 173 173 174 174 174 174 174 175 175 175 175 175 176 176
176 176 176 177 177 177 177 177 178 178 178 178 178 179 179 179 179 179
180 180 180 180 180 181 181 181 181 181 182 182 182 182 182 183 183 183
183 183 184 184 184 184 184 185 185 185 185 185 186 186 186 186 186 187
187 187 187 187 188 188 188 188 188 189 189 189 189 189 190 190 190 190
190 191 191 191 191 191 192 192 192 192 192 193 193 193 193 193 194 194
194 194 194 195 195 195 195 195 196 196 196 196 196 197 197 197 197 197
198 198 198 198 198 199 199 199 199 199] !

2020-11-26 15:08:16,733 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/IdSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> senders:
[  0   0   0   0   0   1   1   1   1   1   2   2   2   2   2   3   3   3
   3   3   4   4   4   4   4   5   5   5   5   5   6   6   6   6   6   7
   7   7   7   7   8   8   8   8   8   9   9   9   9   9  10  10  10  10
  10  11  11  11  11  11  12  12  12  12  12  13  13  13  13  13  14  14
  14  14  14  15  15  15  15  15  16  16  16  16  16  17  17  17  17  17
  18  18  18  18  18  19  19  19  19  19  20  20  20  20  20  21  21  21
  21  21  22  22  22  22  22  23  23  23  23  23  24  24  24  24  24  25
  25  25  25  25  26  26  26  26  26  27  27  27  27  27  28  28  28  28
  28  29  29  29  29  29  30  30  30  30  30  31  31  31  31  31  32  32
  32  32  32  33  33  33  33  33  34  34  34  34  34  35  35  35  35  35
  36  36  36  36  36  37  37  37  37  37  38  38  38  38  38  39  39  39
  39  39  40  40  40  40  40  41  41  41  41  41  42  42  42  42  42  43
  43  43  43  43  44  44  44  44  44  45  45  45  45  45  46  46  46  46
  46  47  47  47  47  47  48  48  48  48  48  49  49  49  49  49  50  50
  50  50  50  51  51  51  51  51  52  52  52  52  52  53  53  53  53  53
  54  54  54  54  54  55  55  55  55  55  56  56  56  56  56  57  57  57
  57  57  58  58  58  58  58  59  59  59  59  59  60  60  60  60  60  61
  61  61  61  61  62  62  62  62  62  63  63  63  63  63  64  64  64  64
  64  65  65  65  65  65  66  66  66  66  66  67  67  67  67  67  68  68
  68  68  68  69  69  69  69  69  70  70  70  70  70  71  71  71  71  71
  72  72  72  72  72  73  73  73  73  73  74  74  74  74  74  75  75  75
  75  75  76  76  76  76  76  77  77  77  77  77  78  78  78  78  78  79
  79  79  79  79  80  80  80  80  80  81  81  81  81  81  82  82  82  82
  82  83  83  83  83  83  84  84  84  84  84  85  85  85  85  85  86  86
  86  86  86  87  87  87  87  87  88  88  88  88  88  89  89  89  89  89
  90  90  90  90  90  91  91  91  91  91  92  92  92  92  92  93  93  93
  93  93  94  94  94  94  94  95  95  95  95  95  96  96  96  96  96  97
  97  97  97  97  98  98  98  98  98  99  99  99  99  99 100 100 100 100
 100 101 101 101 101 101 102 102 102 102 102 103 103 103 103 103 104 104
 104 104 104 105 105 105 105 105 106 106 106 106 106 107 107 107 107 107
 108 108 108 108 108 109 109 109 109 109 110 110 110 110 110 111 111 111
 111 111 112 112 112 112 112 113 113 113 113 113 114 114 114 114 114 115
 115 115 115 115 116 116 116 116 116 117 117 117 117 117 118 118 118 118
 118 119 119 119 119 119 120 120 120 120 120 121 121 121 121 121 122 122
 122 122 122 123 123 123 123 123 124 124 124 124 124 125 125 125 125 125
```

```
 126 126 126 126 126 127 127 127 127 127 128 128 128 128 128 129 129 129
 129 129 130 130 130 130 130 131 131 131 131 131 132 132 132 132 132 133
 133 133 133 133 134 134 134 134 134 135 135 135 135 135 136 136 136 136
 136 137 137 137 137 137 138 138 138 138 138 139 139 139 139 139 140 140
 140 140 140 141 141 141 141 141 142 142 142 142 142 143 143 143 143 143
 144 144 144 144 144 145 145 145 145 145 146 146 146 146 146 147 147 147
 147 147 148 148 148 148 148 149 149 149 149 149 150 150 150 150 150 151
 151 151 151 151 152 152 152 152 152 153 153 153 153 153 154 154 154 154
 154 155 155 155 155 155 156 156 156 156 156 157 157 157 157 157 158 158
 158 158 158 159 159 159 159 159 160 160 160 160 160 161 161 161 161 161
 162 162 162 162 162 163 163 163 163 163 164 164 164 164 164 165 165 165
 165 165 166 166 166 166 166 167 167 167 167 167 168 168 168 168 168 169
 169 169 169 169 170 170 170 170 170 171 171 171 171 171 172 172 172 172
 172 173 173 173 173 173 174 174 174 174 174 175 175 175 175 175 176 176
 176 176 176 177 177 177 177 177 178 178 178 178 178 179 179 179 179 179
 180 180 180 180 180 181 181 181 181 181 182 182 182 182 182 183 183 183
 183 183 184 184 184 184 184 185 185 185 185 185 186 186 186 186 186 187
 187 187 187 187 188 188 188 188 188 189 189 189 189 189 190 190 190 190
 190 191 191 191 191 191 192 192 192 192 192 193 193 193 193 193 194 194
 194 194 194 195 195 195 195 195 196 196 196 196 196 197 197 197 197 197
 198 198 198 198 198 199 199 199 199 199] !

2020-11-26 15:08:16,745 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,745 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> times to h5 file as a numpy array dataset !

2020-11-26 15:08:16,753 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,753 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> senders to h5 file as a numpy array dataset !

2020-11-26 15:08:16,806 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/IiSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> times:
[  4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
```

```
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
 101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275
  73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65     25.35
  48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2      4.65
  25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9    101.2
   4.65     25.35     48.275    73.9    101.2      4.65     25.35     48.275    73.9
```

```
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2  ] !

2020-11-26 15:08:16,806 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/IiSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> times:
[  4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
  25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275   73.9
 101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35    48.275
  73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65    25.35
  48.275   73.9    101.2     4.65    25.35    48.275   73.9    101.2     4.65
```

```
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275
  73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65    25.35
  48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2      4.65
  25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9    101.2
   4.65    25.35    48.275   73.9    101.2      4.65    25.35    48.275   73.9
 101.2   ] !

2020-11-26 15:08:16,824 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/IiSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> senders:
[  0   0   0   0   0   1   1   1   1   1   2   2   2   2   2   3   3   3
   3   3   4   4   4   4   4   5   5   5   5   5   6   6   6   6   6   7
   7   7   7   7   8   8   8   8   8   9   9   9   9   9  10  10  10  10
  10  11  11  11  11  11  12  12  12  12  12  13  13  13  13  13  14  14
  14  14  14  15  15  15  15  15  16  16  16  16  16  17  17  17  17  17
  18  18  18  18  18  19  19  19  19  19  20  20  20  20  20  21  21  21
  21  21  22  22  22  22  22  23  23  23  23  23  24  24  24  24  24  25
```

```
  25  25  25  25  26  26  26  26  26  27  27  27  27  27  28  28  28  28
  28  29  29  29  29  29  30  30  30  30  30  31  31  31  31  31  32  32
  32  32  32  33  33  33  33  33  34  34  34  34  34  35  35  35  35  35
  36  36  36  36  36  37  37  37  37  37  38  38  38  38  38  39  39  39
  39  39  40  40  40  40  40  41  41  41  41  41  42  42  42  42  42  43
  43  43  43  43  44  44  44  44  44  45  45  45  45  45  46  46  46  46
  46  47  47  47  47  47  48  48  48  48  48  49  49  49  49  49  50  50
  50  50  50  51  51  51  51  51  52  52  52  52  52  53  53  53  53  53
  54  54  54  54  54  55  55  55  55  55  56  56  56  56  56  57  57  57
  57  57  58  58  58  58  58  59  59  59  59  59  60  60  60  60  60  61
  61  61  61  61  62  62  62  62  62  63  63  63  63  63  64  64  64  64
  64  65  65  65  65  65  66  66  66  66  66  67  67  67  67  67  68  68
  68  68  68  69  69  69  69  69  70  70  70  70  70  71  71  71  71  71
  72  72  72  72  72  73  73  73  73  73  74  74  74  74  74  75  75  75
  75  75  76  76  76  76  76  77  77  77  77  77  78  78  78  78  78  79
  79  79  79  79  80  80  80  80  80  81  81  81  81  81  82  82  82  82
  82  83  83  83  83  83  84  84  84  84  84  85  85  85  85  85  86  86
  86  86  86  87  87  87  87  87  88  88  88  88  88  89  89  89  89  89
  90  90  90  90  90  91  91  91  91  91  92  92  92  92  92  93  93  93
  93  93  94  94  94  94  94  95  95  95  95  95  96  96  96  96  96  97
  97  97  97  97  98  98  98  98  98  99  99  99  99  99 100 100 100 100
 100 101 101 101 101 101 102 102 102 102 102 103 103 103 103 103 104 104
 104 104 104 105 105 105 105 105 106 106 106 106 106 107 107 107 107 107
 108 108 108 108 108 109 109 109 109 109 110 110 110 110 110 111 111 111
 111 111 112 112 112 112 112 113 113 113 113 113 114 114 114 114 114 115
 115 115 115 115 116 116 116 116 116 117 117 117 117 117 118 118 118 118
 118 119 119 119 119 119 120 120 120 120 120 121 121 121 121 121 122 122
 122 122 122 123 123 123 123 123 124 124 124 124 124 125 125 125 125 125
 126 126 126 126 126 127 127 127 127 127 128 128 128 128 128 129 129 129
 129 129 130 130 130 130 130 131 131 131 131 131 132 132 132 132 132 133
 133 133 133 133 134 134 134 134 134 135 135 135 135 135 136 136 136 136
 136 137 137 137 137 137 138 138 138 138 138 139 139 139 139 139 140 140
 140 140 140 141 141 141 141 141 142 142 142 142 142 143 143 143 143 143
 144 144 144 144 144 145 145 145 145 145 146 146 146 146 146 147 147 147
 147 147 148 148 148 148 148 149 149 149 149 149 150 150 150 150 150 151
 151 151 151 151 152 152 152 152 152 153 153 153 153 153 154 154 154 154
 154 155 155 155 155 155 156 156 156 156 156 157 157 157 157 157 158 158
 158 158 158 159 159 159 159 159 160 160 160 160 160 161 161 161 161 161
 162 162 162 162 162 163 163 163 163 163 164 164 164 164 164 165 165 165
 165 165 166 166 166 166 166 167 167 167 167 167 168 168 168 168 168 169
 169 169 169 169 170 170 170 170 170 171 171 171 171 171 172 172 172 172
 172 173 173 173 173 173 174 174 174 174 174 175 175 175 175 175 176 176
 176 176 176 177 177 177 177 177 178 178 178 178 178 179 179 179 179 179
 180 180 180 180 180 181 181 181 181 181 182 182 182 182 182 183 183 183
 183 183 184 184 184 184 184 185 185 185 185 185 186 186 186 186 186 187
 187 187 187 187 188 188 188 188 188 189 189 189 189 189 190 190 190 190
 190 191 191 191 191 191 192 192 192 192 192 193 193 193 193 193 194 194
 194 194 194 195 195 195 195 195 196 196 196 196 196 197 197 197 197 197
 198 198 198 198 198 199 199 199 199 199] !

2020-11-26 15:08:16,824 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/IiSN_spikes/Striatum_Left" (2 members)> datas
et <class 'numpy.ndarray'> senders:
[  0   0   0   0   0   1   1   1   1   1   2   2   2   2   2   3   3   3
   3   3   4   4   4   4   4   5   5   5   5   5   6   6   6   6   6   7
   7   7   7   7   8   8   8   8   8   9   9   9   9   9  10  10  10  10
  10  11  11  11  11  11  12  12  12  12  12  13  13  13  13  13  14  14
  14  14  14  15  15  15  15  15  16  16  16  16  16  17  17  17  17  17
  18  18  18  18  18  19  19  19  19  19  20  20  20  20  20  21  21  21
  21  21  22  22  22  22  22  23  23  23  23  23  24  24  24  24  24  25
  25  25  25  25  26  26  26  26  26  27  27  27  27  27  28  28  28  28
  28  29  29  29  29  29  30  30  30  30  30  31  31  31  31  31  32  32
  32  32  32  33  33  33  33  33  34  34  34  34  34  35  35  35  35  35
  36  36  36  36  36  37  37  37  37  37  38  38  38  38  38  39  39  39
  39  39  40  40  40  40  40  41  41  41  41  41  42  42  42  42  42  43
  43  43  43  43  44  44  44  44  44  45  45  45  45  45  46  46  46  46
  46  47  47  47  47  47  48  48  48  48  48  49  49  49  49  49  50  50
  50  50  50  51  51  51  51  51  52  52  52  52  52  53  53  53  53  53
  54  54  54  54  54  55  55  55  55  55  56  56  56  56  56  57  57  57
```

```
  57  57  58  58  58  58  58  59  59  59  59  59  60  60  60  60  60  61
  61  61  61  61  62  62  62  62  62  63  63  63  63  63  64  64  64  64
  64  65  65  65  65  65  66  66  66  66  66  67  67  67  67  67  68  68
  68  68  68  69  69  69  69  69  70  70  70  70  70  71  71  71  71  71
  72  72  72  72  72  73  73  73  73  73  74  74  74  74  74  75  75  75
  75  75  76  76  76  76  76  77  77  77  77  77  78  78  78  78  78  79
  79  79  79  79  80  80  80  80  80  81  81  81  81  81  82  82  82  82
  82  83  83  83  83  83  84  84  84  84  84  85  85  85  85  85  86  86
  86  86  86  87  87  87  87  87  88  88  88  88  88  89  89  89  89  89
  90  90  90  90  90  91  91  91  91  91  92  92  92  92  92  93  93  93
  93  93  94  94  94  94  94  95  95  95  95  95  96  96  96  96  96  97
  97  97  97  97  98  98  98  98  98  99  99  99  99  99 100 100 100 100
 100 101 101 101 101 101 102 102 102 102 102 103 103 103 103 103 104 104
 104 104 104 105 105 105 105 105 106 106 106 106 106 107 107 107 107 107
 108 108 108 108 108 109 109 109 109 109 110 110 110 110 110 111 111 111
 111 111 112 112 112 112 112 113 113 113 113 113 114 114 114 114 114 115
 115 115 115 115 116 116 116 116 116 117 117 117 117 117 118 118 118 118
 118 119 119 119 119 119 120 120 120 120 120 121 121 121 121 121 122 122
 122 122 122 123 123 123 123 123 124 124 124 124 124 125 125 125 125 125
 126 126 126 126 126 127 127 127 127 127 128 128 128 128 128 129 129 129
 129 129 130 130 130 130 130 131 131 131 131 131 132 132 132 132 132 133
 133 133 133 133 134 134 134 134 134 135 135 135 135 135 136 136 136 136
 136 137 137 137 137 137 138 138 138 138 138 139 139 139 139 139 140 140
 140 140 140 141 141 141 141 141 142 142 142 142 142 143 143 143 143 143
 144 144 144 144 144 145 145 145 145 145 146 146 146 146 146 147 147 147
 147 147 148 148 148 148 148 149 149 149 149 149 150 150 150 150 150 151
 151 151 151 151 152 152 152 152 152 153 153 153 153 153 154 154 154 154
 154 155 155 155 155 155 156 156 156 156 156 157 157 157 157 157 158 158
 158 158 158 159 159 159 159 159 160 160 160 160 160 161 161 161 161 161
 162 162 162 162 162 163 163 163 163 163 164 164 164 164 164 165 165 165
 165 165 166 166 166 166 166 167 167 167 167 167 168 168 168 168 168 169
 169 169 169 169 170 170 170 170 170 171 171 171 171 171 172 172 172 172
 172 173 173 173 173 173 174 174 174 174 174 175 175 175 175 175 176 176
 176 176 176 177 177 177 177 177 178 178 178 178 178 179 179 179 179 179
 180 180 180 180 180 181 181 181 181 181 182 182 182 182 182 183 183 183
 183 183 184 184 184 184 184 185 185 185 185 185 186 186 186 186 186 187
 187 187 187 187 188 188 188 188 188 189 189 189 189 189 190 190 190 190
 190 191 191 191 191 191 192 192 192 192 192 193 193 193 193 193 194 194
 194 194 194 195 195 195 195 195 196 196 196 196 196 197 197 197 197 197
 198 198 198 198 198 199 199 199 199 199] !

2020-11-26 15:08:16,867 - INFO - tvb_multiscale.core.io.h5_writer - dict has b
een written to file: /home/docker/packages/tvb-multiscale/examples/notebooks/o
utputs_Izhikevich_annarchy/res/Spikes.h5
2020-11-26 15:08:16,867 - INFO - tvb_multiscale.core.io.h5_writer - dict has b
een written to file: /home/docker/packages/tvb-multiscale/examples/notebooks/o
utputs_Izhikevich_annarchy/res/Spikes.h5
2020-11-26 15:08:16,877 - INFO - tvb_multiscale.core.io.h5_writer - Starting t
o write dict to: /home/docker/packages/tvb-multiscale/examples/notebooks/outpu
ts_Izhikevich_annarchy/res/Mean Populations' Spikes' Rates.h5
2020-11-26 15:08:16,877 - INFO - tvb_multiscale.core.io.h5_writer - Starting t
o write dict to: /home/docker/packages/tvb-multiscale/examples/notebooks/outpu
ts_Izhikevich_annarchy/res/Mean Populations' Spikes' Rates.h5
2020-11-26 15:08:16,897 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:16,897 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:16,905 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:16,905 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:16,913 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Mean Populations' Spikes' Rates.h5" (mode r+)>
dataset <class 'numpy.ndarray'> dims:
['Population' 'Region'] !
```

```
2020-11-26 15:08:16,913 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Mean Populations' Spikes' Rates.h5" (mode r+)>
dataset <class 'numpy.ndarray'> dims:
['Population' 'Region'] !

2020-11-26 15:08:16,925 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Mean Populations' Spikes' Rates.h5" (mode r+)>
dataset <class 'numpy.ndarray'> data:
[[24.24242424         nan         nan 16.16161616         nan  0.        ]
 [12.12121212 14.14141414 10.1010101          nan         nan         nan]
 [        nan         nan         nan         nan  8.08080808         nan]
 [        nan         nan         nan         nan  8.08080808         nan]] !

2020-11-26 15:08:16,925 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Mean Populations' Spikes' Rates.h5" (mode r+)>
dataset <class 'numpy.ndarray'> data:
[[24.24242424         nan         nan 16.16161616         nan  0.        ]
 [12.12121212 14.14141414 10.1010101          nan         nan         nan]
 [        nan         nan         nan         nan  8.08080808         nan]
 [        nan         nan         nan         nan  8.08080808         nan]] !

2020-11-26 15:08:16,941 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:16,941 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:16,948 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:16,948 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:16,957 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region" (2 members)> dataset <class 'n
umpy.ndarray'> dims:
['Region'] !

2020-11-26 15:08:16,957 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region" (2 members)> dataset <class 'n
umpy.ndarray'> dims:
['Region'] !

2020-11-26 15:08:16,967 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region" (2 members)> dataset <class 'n
umpy.ndarray'> data:
['Cortex' 'GPe_Left' 'GPi_Left' 'STN_Left' 'Striatum_Left' 'Thal_Left'] !

2020-11-26 15:08:16,967 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region" (2 members)> dataset <class 'n
umpy.ndarray'> data:
['Cortex' 'GPe_Left' 'GPi_Left' 'STN_Left' 'Striatum_Left' 'Thal_Left'] !

2020-11-26 15:08:16,980 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:16,980 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:16,988 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:16,988 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:16,996 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population" (2 members)> dataset <clas
s 'numpy.ndarray'> dims:
```

```
['Population'] !

2020-11-26 15:08:16,996 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population" (2 members)> dataset <clas
s 'numpy.ndarray'> dims:
['Population'] !

2020-11-26 15:08:17,005 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population" (2 members)> dataset <clas
s 'numpy.ndarray'> data:
['E_spikes' 'I_spikes' 'IdSN_spikes' 'IiSN_spikes'] !

2020-11-26 15:08:17,005 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population" (2 members)> dataset <clas
s 'numpy.ndarray'> data:
['E_spikes' 'I_spikes' 'IdSN_spikes' 'IiSN_spikes'] !

2020-11-26 15:08:17,026 - INFO - tvb_multiscale.core.io.h5_writer - dict has b
een written to file: /home/docker/packages/tvb-multiscale/examples/notebooks/o
utputs_Izhikevich_annarchy/res/Mean Populations' Spikes' Rates.h5
2020-11-26 15:08:17,026 - INFO - tvb_multiscale.core.io.h5_writer - dict has b
een written to file: /home/docker/packages/tvb-multiscale/examples/notebooks/o
utputs_Izhikevich_annarchy/res/Mean Populations' Spikes' Rates.h5
2020-11-26 15:08:17,261 - INFO - tvb_multiscale.core.io.h5_writer - Starting t
o write dict to: /home/docker/packages/tvb-multiscale/examples/notebooks/outpu
ts_Izhikevich_annarchy/res/Populations' Correlation Coefficient.h5
2020-11-26 15:08:17,261 - INFO - tvb_multiscale.core.io.h5_writer - Starting t
o write dict to: /home/docker/packages/tvb-multiscale/examples/notebooks/outpu
ts_Izhikevich_annarchy/res/Populations' Correlation Coefficient.h5
2020-11-26 15:08:17,277 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,277 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,284 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,284 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,293 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Populations' Correlation Coefficient.h5" (mode
r+)> dataset <class 'numpy.ndarray'> dims:
['Population_i' 'Population_j' 'Region_i' 'Region_j'] !

2020-11-26 15:08:17,293 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Populations' Correlation Coefficient.h5" (mode
r+)> dataset <class 'numpy.ndarray'> dims:
['Population_i' 'Population_j' 'Region_i' 'Region_j'] !

2020-11-26 15:08:17,350 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Populations' Correlation Coefficient.h5" (mode
 r+)> dataset <class 'numpy.ndarray'> data:
[[[[nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]]

  [[nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]]

  [[nan nan nan nan nan nan]
```

```
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]]

      [[nan nan nan nan nan nan]
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]
       [nan nan nan nan nan nan]]]


    [[[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]

     [[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]

     [[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]

     [[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]]


    [[[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]

     [[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]

     [[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]

     [[nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]
      [nan nan nan nan nan nan]]
```

```
    [nan nan nan nan nan nan]]]


  [[[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]

   [[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]

   [[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]

   [[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]]] !

2020-11-26 15:08:17,350 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 file "Populations' Correlation Coefficient.h5" (mode
 r+)> dataset <class 'numpy.ndarray'> data:
[[[[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]

   [[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]

   [[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]

   [[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]]


  [[[nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]
    [nan nan nan nan nan nan]]
```

```
 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]

 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]

 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]]


[[[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]

 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]

 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]

 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]]


[[[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]

 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]]

 [[nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
  [nan nan nan nan nan nan]
```

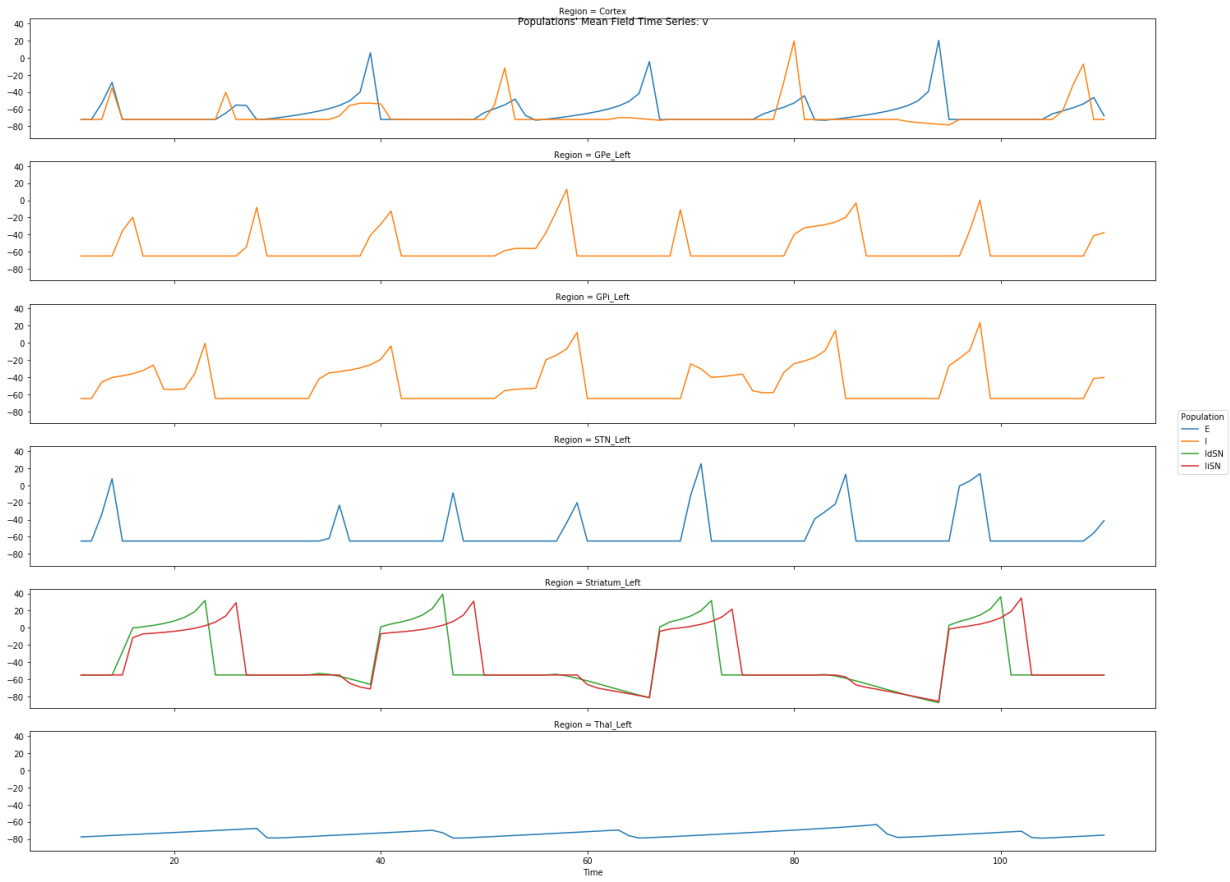
```
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]]

  [[nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]
   [nan nan nan nan nan nan]]]] !

2020-11-26 15:08:17,362 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,362 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,371 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,371 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,380 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_i" (2 members)> dataset <cl
ass 'numpy.ndarray'> dims:
['Population_i'] !

2020-11-26 15:08:17,380 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_i" (2 members)> dataset <cl
ass 'numpy.ndarray'> dims:
['Population_i'] !

2020-11-26 15:08:17,388 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_i" (2 members)> dataset <cl
ass 'numpy.ndarray'> data:
['E_spikes' 'I_spikes' 'IdSN_spikes' 'IiSN_spikes'] !

2020-11-26 15:08:17,388 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_i" (2 members)> dataset <cl
ass 'numpy.ndarray'> data:
['E_spikes' 'I_spikes' 'IdSN_spikes' 'IiSN_spikes'] !

2020-11-26 15:08:17,404 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,404 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,410 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,410 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,417 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_i" (2 members)> dataset <class
 'numpy.ndarray'> dims:
['Region_i'] !

2020-11-26 15:08:17,417 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_i" (2 members)> dataset <class
 'numpy.ndarray'> dims:
['Region_i'] !

2020-11-26 15:08:17,429 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_i" (2 members)> dataset <class
 'numpy.ndarray'> data:
['Cortex' 'GPe_Left' 'GPi_Left' 'STN_Left' 'Striatum_Left' 'Thal_Left'] !
```

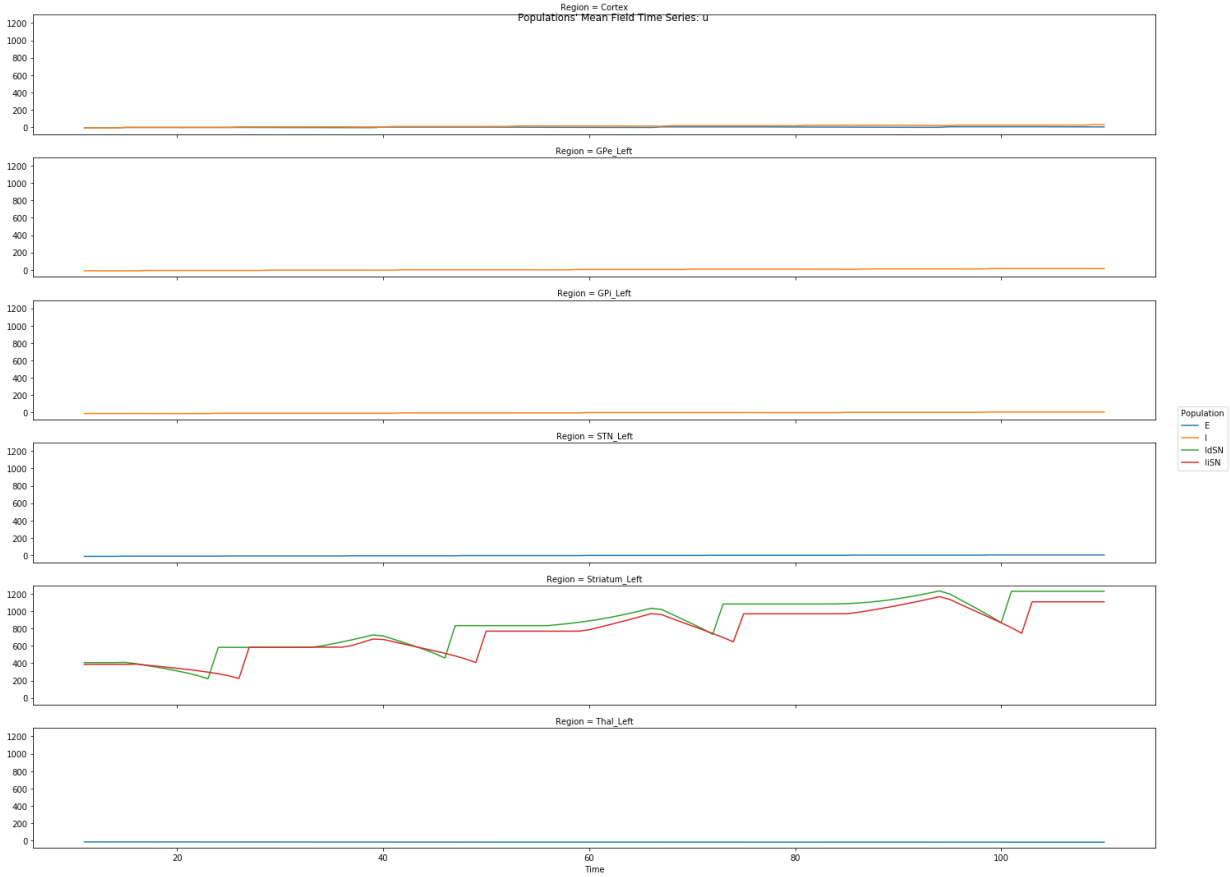
```
2020-11-26 15:08:17,429 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_i" (2 members)> dataset <class
 'numpy.ndarray'> data:
['Cortex' 'GPe_Left' 'GPi_Left' 'STN_Left' 'Striatum_Left' 'Thal_Left'] !

2020-11-26 15:08:17,445 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,445 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,450 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,450 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,466 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_j" (2 members)> dataset <cl
ass 'numpy.ndarray'> dims:
['Population_j'] !

2020-11-26 15:08:17,466 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_j" (2 members)> dataset <cl
ass 'numpy.ndarray'> dims:
['Population_j'] !

2020-11-26 15:08:17,475 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_j" (2 members)> dataset <cl
ass 'numpy.ndarray'> data:
['E_spikes' 'I_spikes' 'IdSN_spikes' 'IiSN_spikes'] !

2020-11-26 15:08:17,475 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Population_j" (2 members)> dataset <cl
ass 'numpy.ndarray'> data:
['E_spikes' 'I_spikes' 'IdSN_spikes' 'IiSN_spikes'] !

2020-11-26 15:08:17,488 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,488 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'tuple'> dims to h5 file as a numpy array dataset !

2020-11-26 15:08:17,494 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,494 - WARNING - tvb_multiscale.core.io.h5_writer -
Writing <class 'list'> data to h5 file as a numpy array dataset !

2020-11-26 15:08:17,501 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_j" (2 members)> dataset <class
 'numpy.ndarray'> dims:
['Region_j'] !

2020-11-26 15:08:17,501 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_j" (2 members)> dataset <class
 'numpy.ndarray'> dims:
['Region_j'] !

2020-11-26 15:08:17,511 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_j" (2 members)> dataset <class
 'numpy.ndarray'> data:
['Cortex' 'GPe_Left' 'GPi_Left' 'STN_Left' 'Striatum_Left' 'Thal_Left'] !

2020-11-26 15:08:17,511 - WARNING - tvb_multiscale.core.io.h5_writer -
Failed to write to <HDF5 group "/coords/Region_j" (2 members)> dataset <class
 'numpy.ndarray'> data:
['Cortex' 'GPe_Left' 'GPi_Left' 'STN_Left' 'Striatum_Left' 'Thal_Left'] !
```

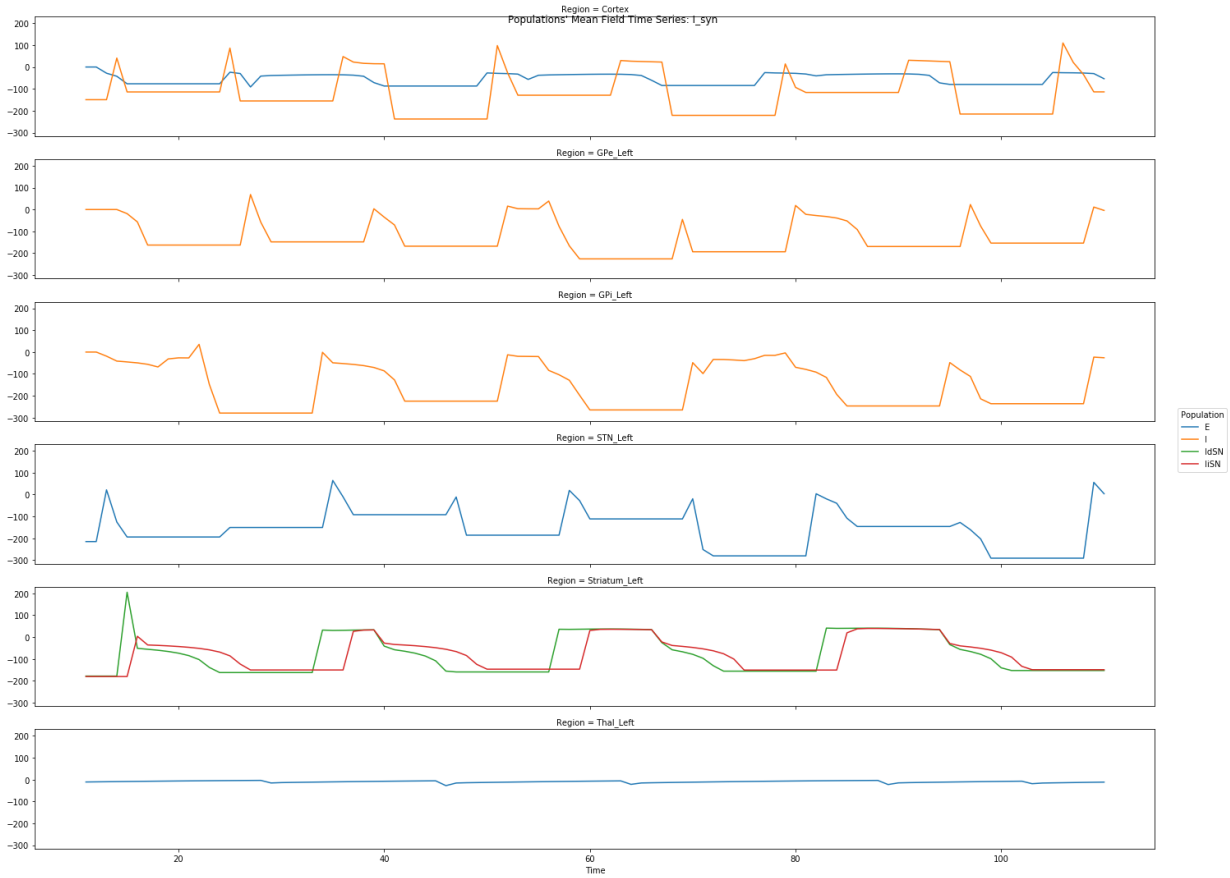
```
2020-11-26 15:08:17,540 - INFO - tvb_multiscale.core.io.h5_writer - dict has b
een written to file: /home/docker/packages/tvb-multiscale/examples/notebooks/o
utputs_Izhikevich_annarchy/res/Populations' Correlation Coefficient.h5
2020-11-26 15:08:17,540 - INFO - tvb_multiscale.core.io.h5_writer - dict has b
een written to file: /home/docker/packages/tvb-multiscale/examples/notebooks/o
utputs_Izhikevich_annarchy/res/Populations' Correlation Coefficient.h5
```

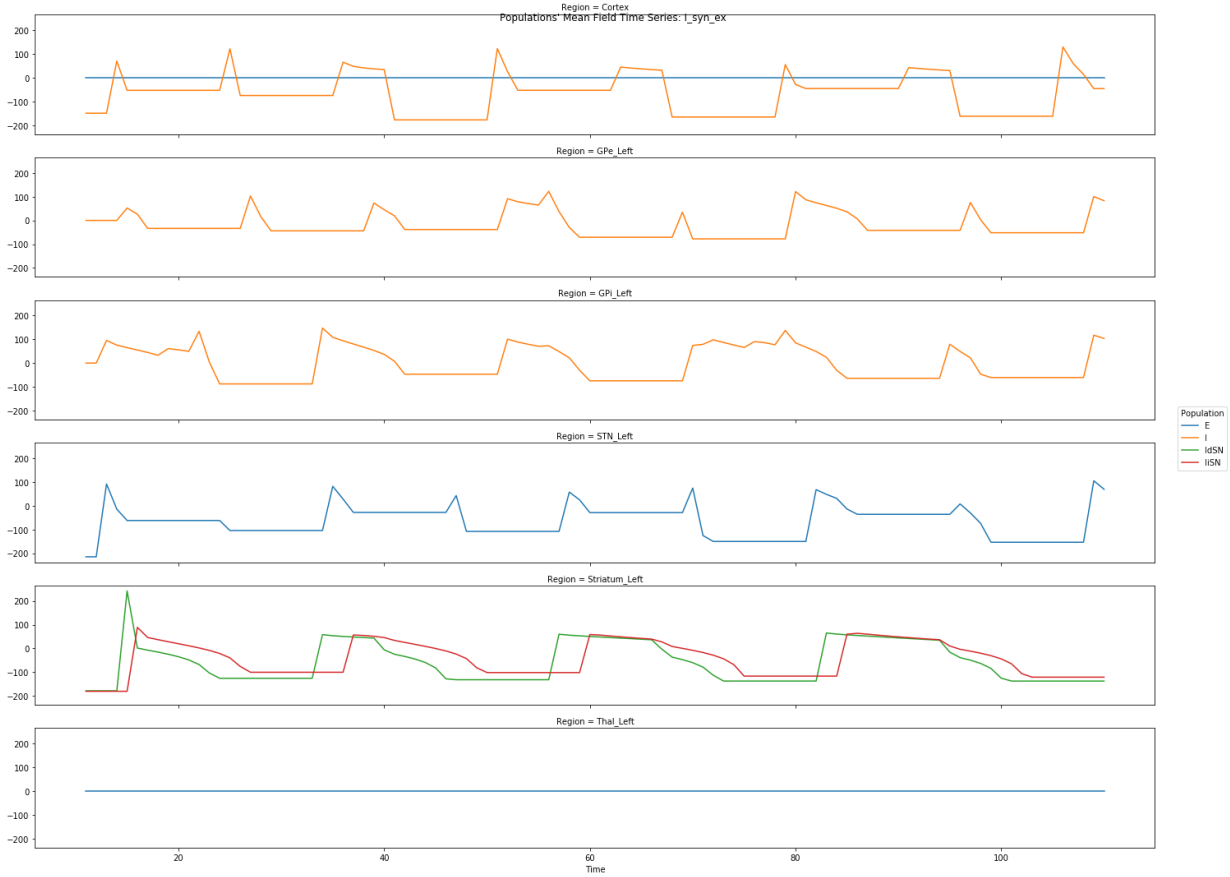
Get SpikingNetwork mean field variable time series and plot them

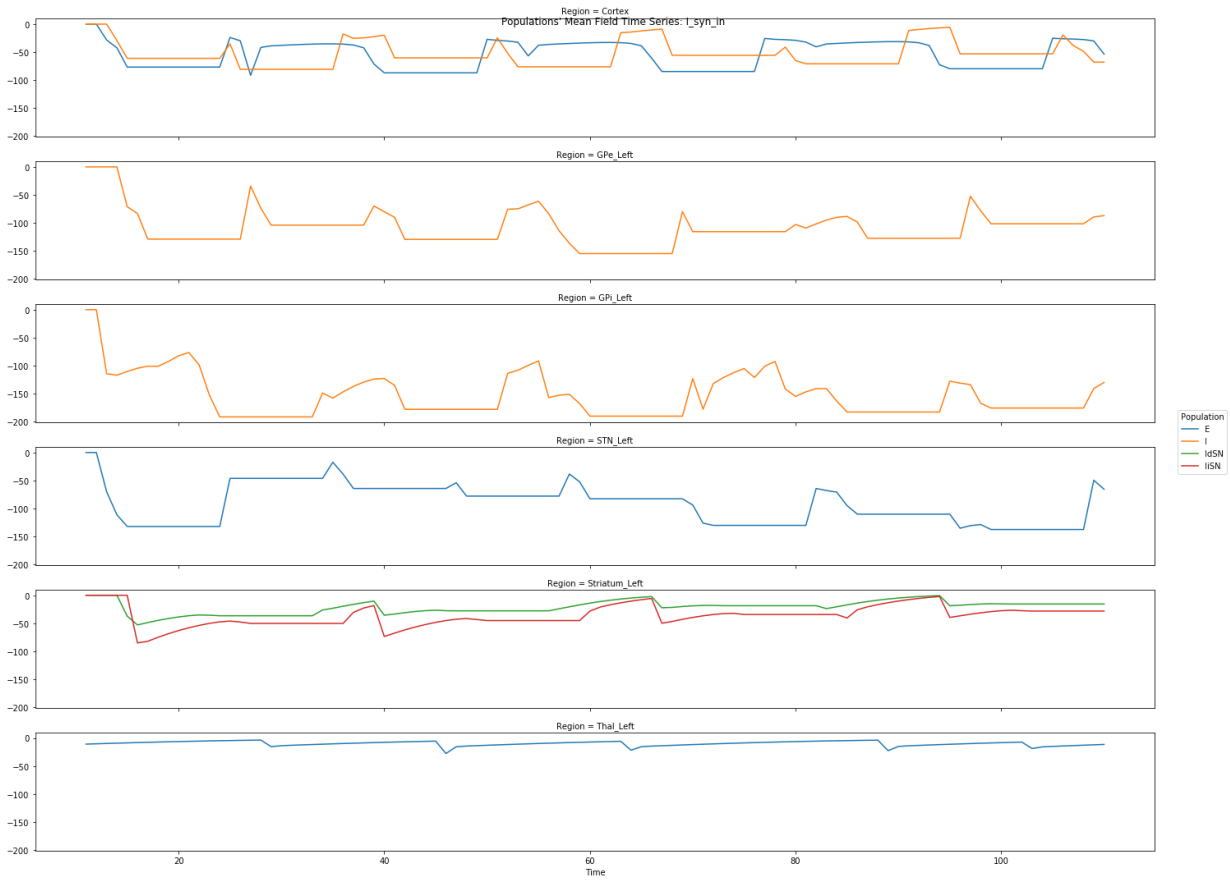
In [12]:
```
# Continuous time variables' data of spiking neurons
if plot_per_neuron:
    spikeNet_analyzer.return_data = True
else:
    spikeNet_analyzer.return_data = False
spikeNet_ts = \
    spikeNet_analyzer. \
         compute_spikeNet_mean_field_time_series(populations_devices=None, re
                                                 computations_kwargs={}, data
if spikeNet_ts:
    if plot_per_neuron:
        mean_field_ts = spikeNet_ts["mean_field_time_series"]  # mean field
        spikeNet_ts = spikeNet_ts["data_by_neuron"]  # per neuron data
    else:
        mean_field_ts = spikeNet_ts
    if mean_field_ts and mean_field_ts.size > 0:
        mean_field_ts.plot_timeseries(plotter_config=plotter.config,
                                      per_variable=mean_field_ts.shape[1] > M
        if mean_field_ts.number_of_labels > MIN_REGIONS_FOR_RASTER_PLOT:
            mean_field_ts.plot_raster(plotter_config=plotter.config,
                                      per_variable=mean_field_ts.shape[1] > M
                                      linestyle="--", alpha=0.5, linewidth=0.
else:
    mean_field_ts = None
```

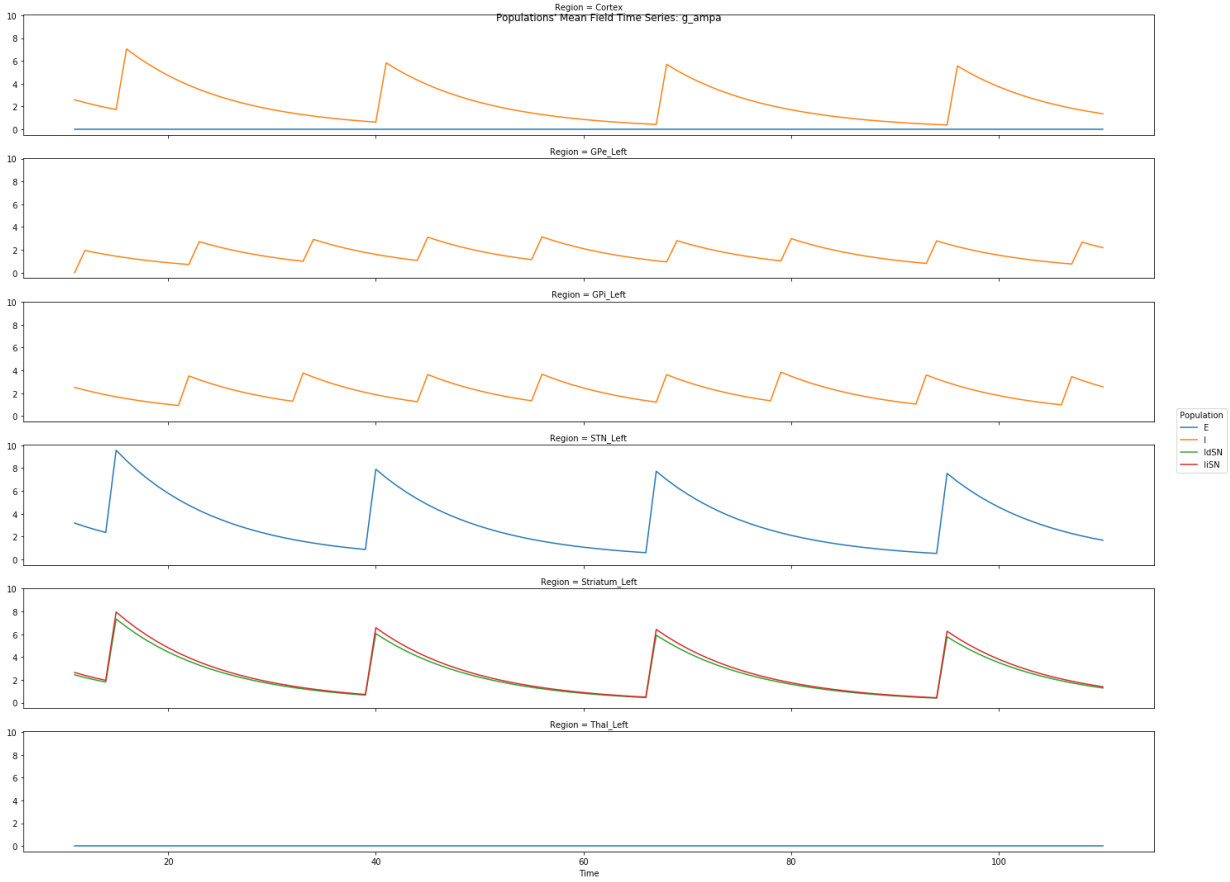


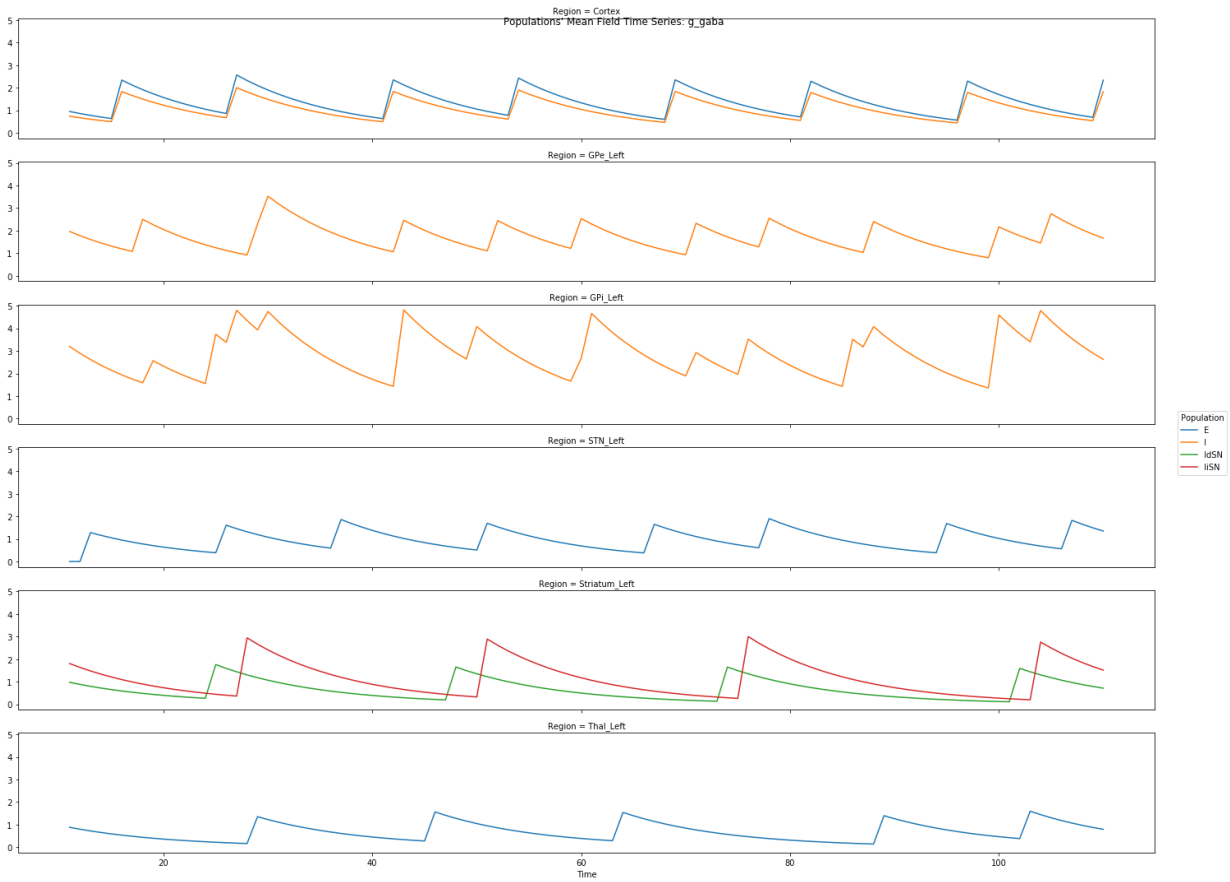

Region = Cortex
Populations' Mean Field Time Series: u
Region = GPe_Left
Region = GPi_Left
Region = STN_Left
Region = Striatum_Left
Region = Thal_Left
Time
Population
E
I
IdSN
IiSN

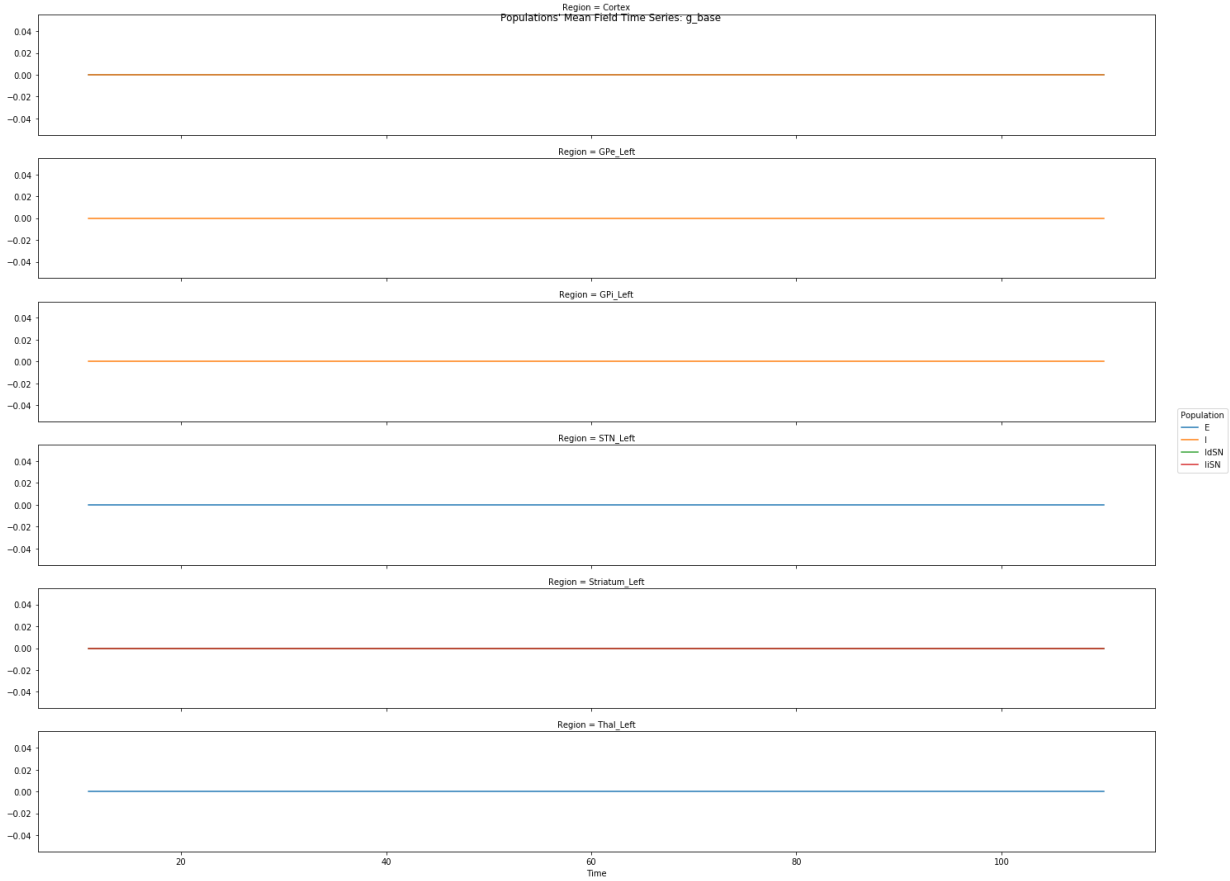

Region = Cortex
Populations' Mean Field Time Series: I_syn
Region = GPe_Left
Region = GPi_Left
Region = STN_Left
Region = Striatum_Left
Region = Thal_Left
Time
Population
E
I
IdSN
IiSN


Region = Cortex
Populations' Mean Field Time Series: I_syn_ex
Region = GPe_Left
Region = GPi_Left
Region = STN_Left
Region = Striatum_Left
Region = Thal_Left
Time
Population
E
I
IdSN
IiSN


Region = Cortex
Populations' Mean Field Time Series: I_syn_in
Region = GPe_Left
Region = GPi_Left
Region = STN_Left
Region = Striatum_Left
Region = Thal_Left
Time
Population
E
I
IdSN
IiSN


Region = Cortex
Populations' Mean Field Time Series: g_ampa
Region = GPe_Left
Region = GPi_Left
Region = STN_Left
Region = Striatum_Left
Region = Thal_Left
Population
E
I
IdSN
IiSN
Time


Region = Cortex
Populations' Mean Field Time Series: g_gaba
Region = GPe_Left
Region = GPi_Left
Region = STN_Left
Region = Striatum_Left
Region = Thal_Left
Population
E
I
IdSN
IiSN
Time

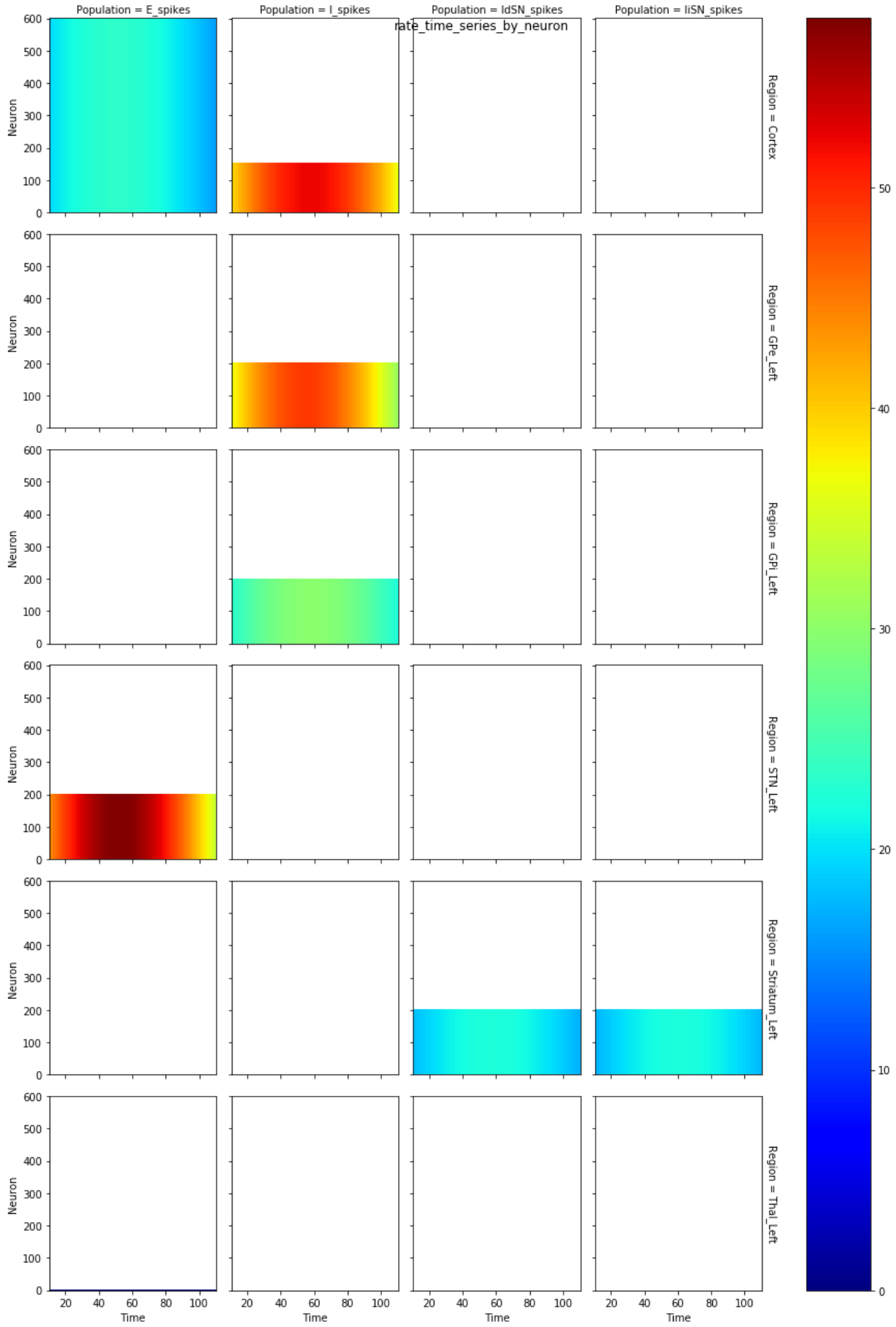


In [13]:
```
# Write results to file:
if mean_field_ts and writer:
    writer.write_tvb_to_h5(TimeSeriesRegion().from_xarray_DataArray(
                                        mean_field_ts._data,
                                        connectivity=mean_field_ts.connectivit
                           os.path.join(config.out.FOLDER_RES, mean_field_ts.
                           recursive=False)
```

Compute per neuron spikes' rates times series and plot them

In [14]:
```
if spikes_res and plot_per_neuron:
    from tvb.simulator.plot.base_plotter import pyplot
    spikeNet_analyzer.return_data = False
    rates_ts_per_neuron = \
        spikeNet_analyzer. \
            compute_spikeNet_rates_time_series(populations_devices=None, regi
                                               computations_kwargs={}, data_k
                                               return_spikes_trains=False, re
    if rates_ts_per_neuron is not None and rates_ts_per_neuron.size:
        # Regions in rows
        row = rates_ts_per_neuron.dims[2] if rates_ts_per_neuron.shape[2] > 1
        if row is None:
            # Populations in rows
            row = rates_ts_per_neuron.dims[1] if rates_ts_per_neuron.shape[1]
            col = None
        else:
            # Populations in columns
            col = rates_ts_per_neuron.dims[1] if rates_ts_per_neuron.shape[1]
        pyplot.figure()
        rates_ts_per_neuron.plot(y=rates_ts_per_neuron.dims[3], row=row, col=
        plotter.base._save_figure(figure_name="Spike rates per neuron")
        # del rates_ts_per_neuron # to free memory
```

```
<Figure size 432x288 with 0 Axes>
```

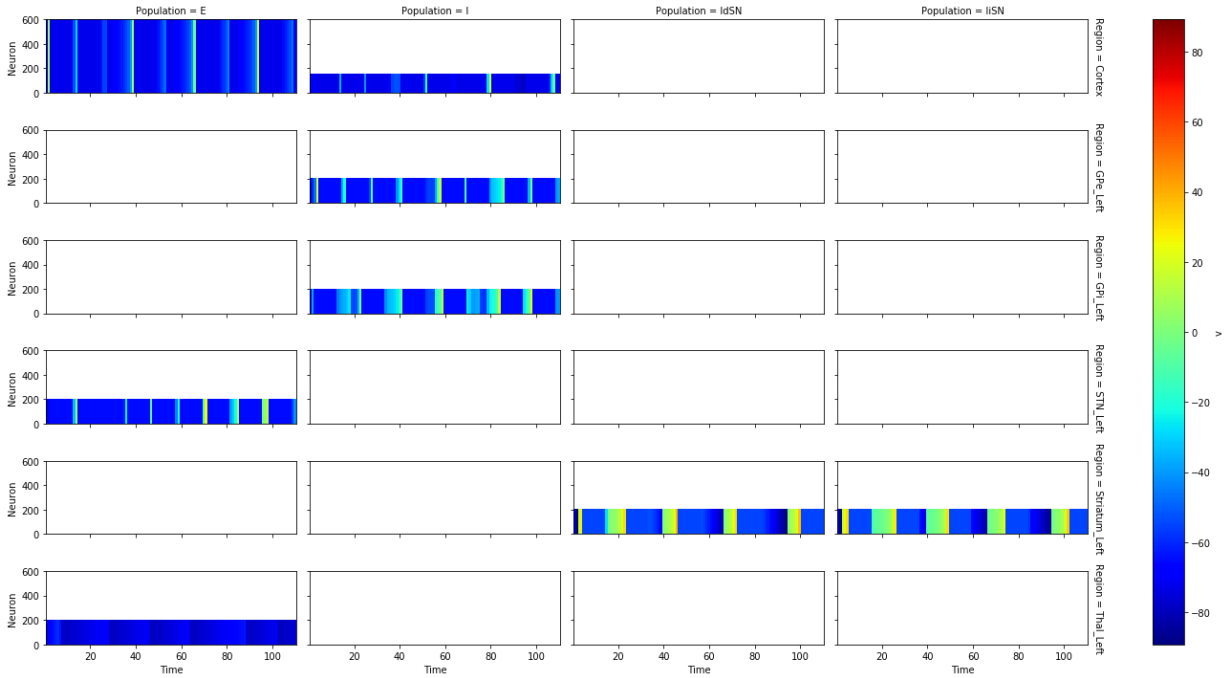


Plot per neuron SpikingNetwork time series

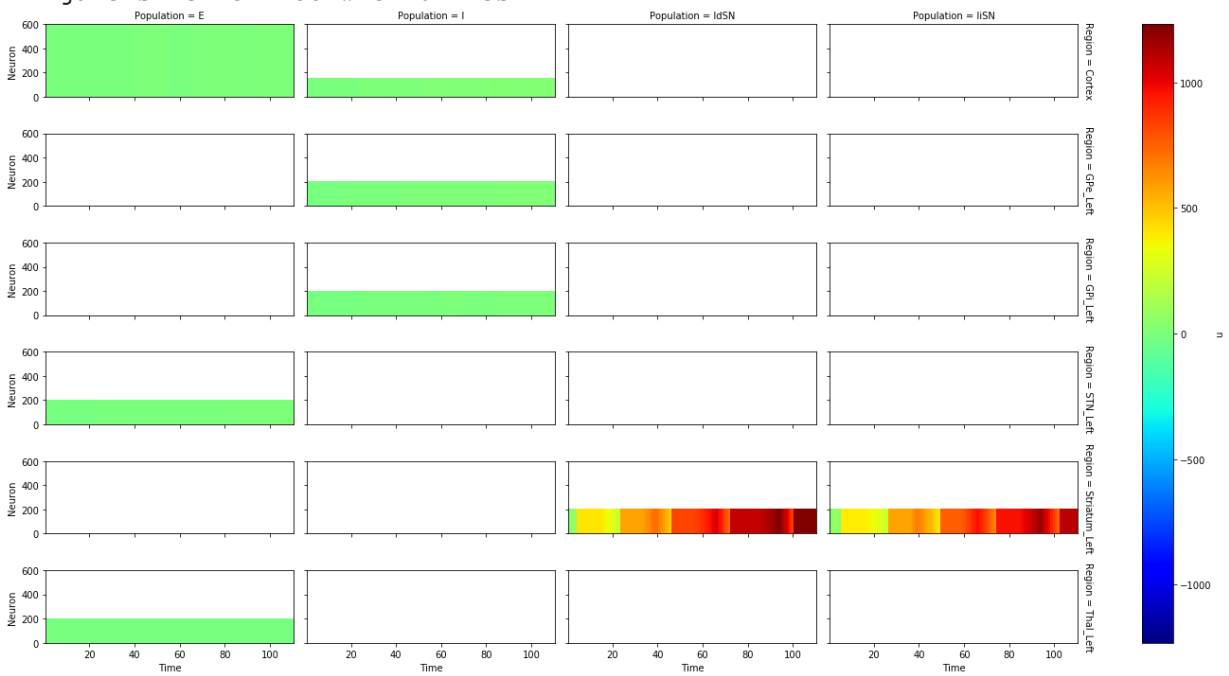
In [15]:
```
# Regions in rows
if plot_per_neuron and spikeNet_ts.size:
    row = spikeNet_ts.dims[2] if spikeNet_ts.shape[2] > 1 else None
    if row is None:
```

```
        # Populations in rows
        row = spikeNet_ts.dims[3] if spikeNet_ts.shape[3] > 1 else None
        col = None
    else:
        # Populations in cols
         col = spikeNet_ts.dims[3] if spikeNet_ts.shape[3] > 1 else None
    for var in spikeNet_ts.coords[spikeNet_ts.dims[1]]:
        this_var_ts = spikeNet_ts.loc[:, var, :, :, :]
        this_var_ts.name = var.item()
        pyplot.figure()
        this_var_ts.plot(y=spikeNet_ts.dims[4], row=row, col=col, cmap="jet",
        plotter.base._save_figure(
            figure_name="Spiking Network variables' time series per neuron: %
    del spikeNet_ts # to free memory
```

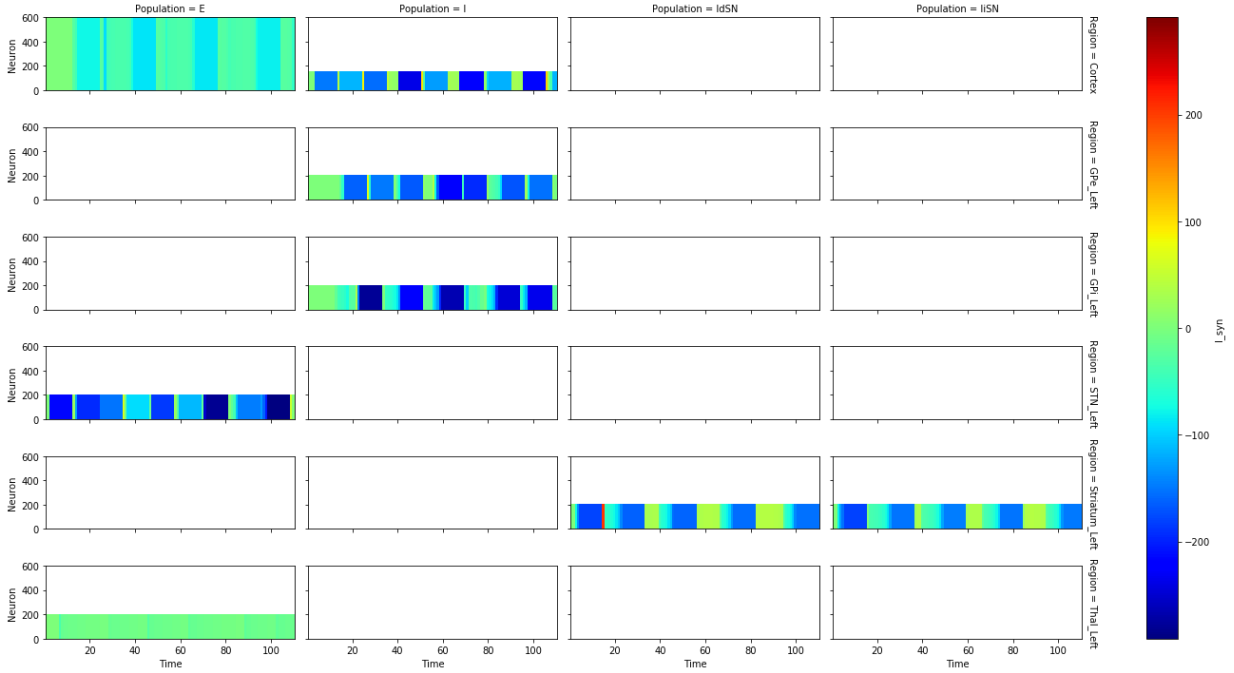
```
<Figure size 432x288 with 0 Axes>
```



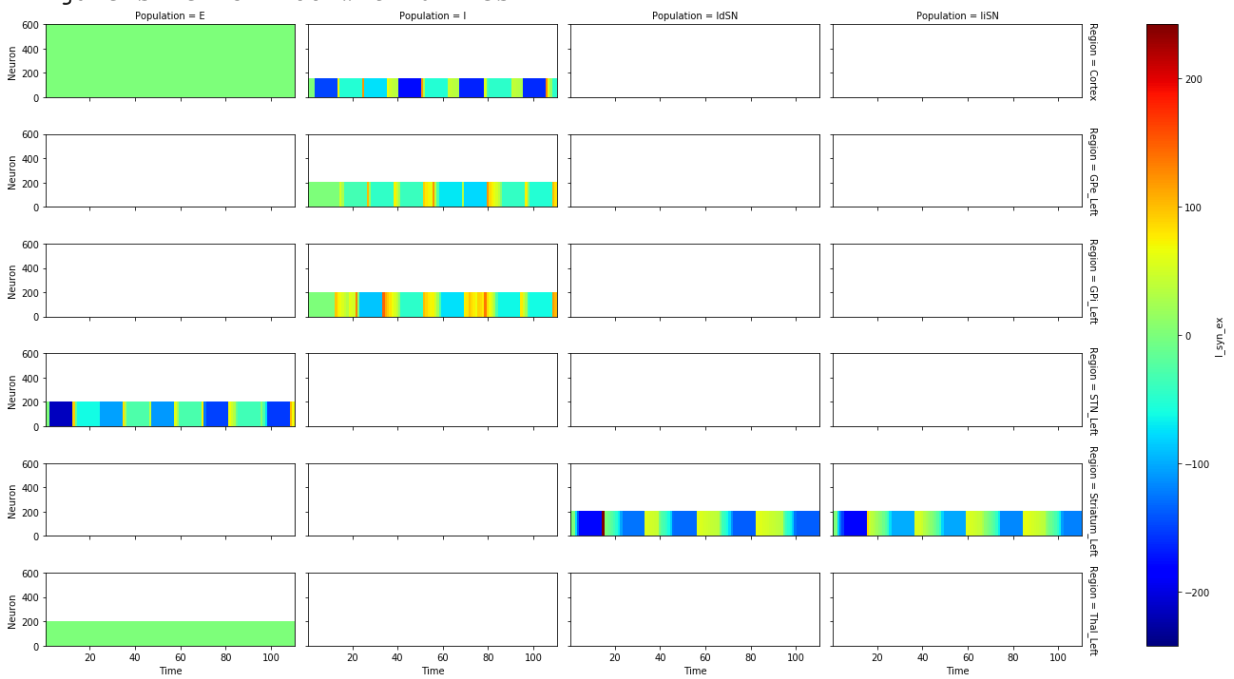

```
<Figure size 432x288 with 0 Axes>
```



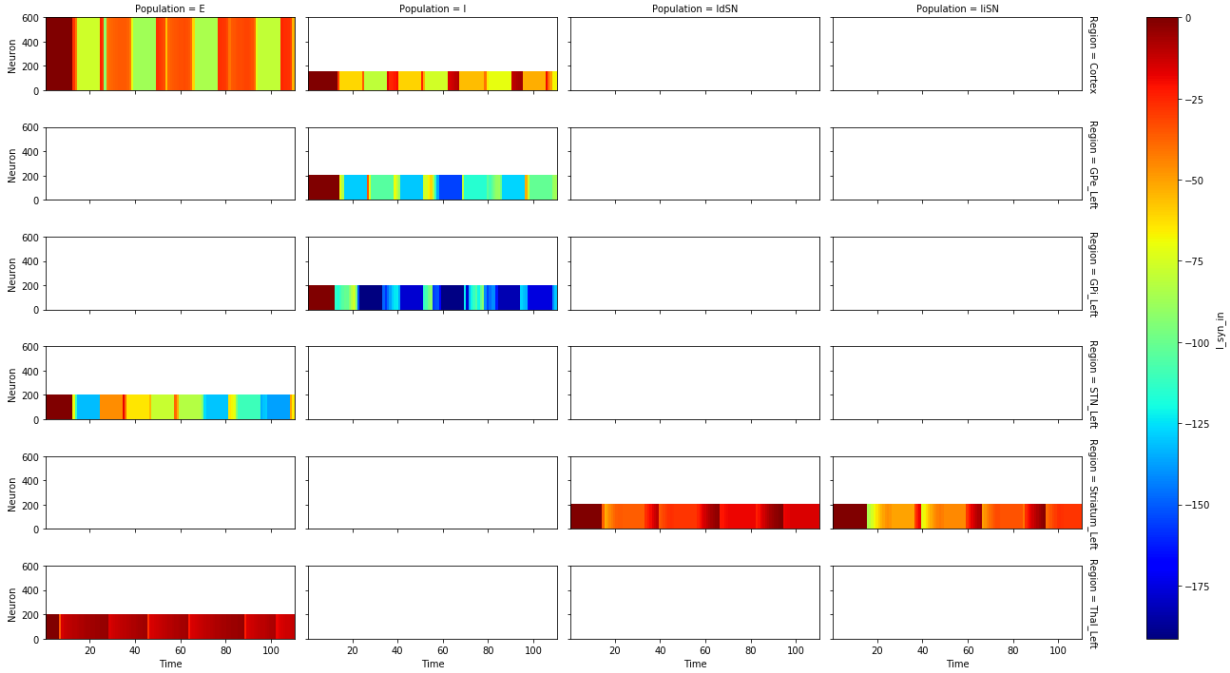

```
<Figure size 432x288 with 0 Axes>
```



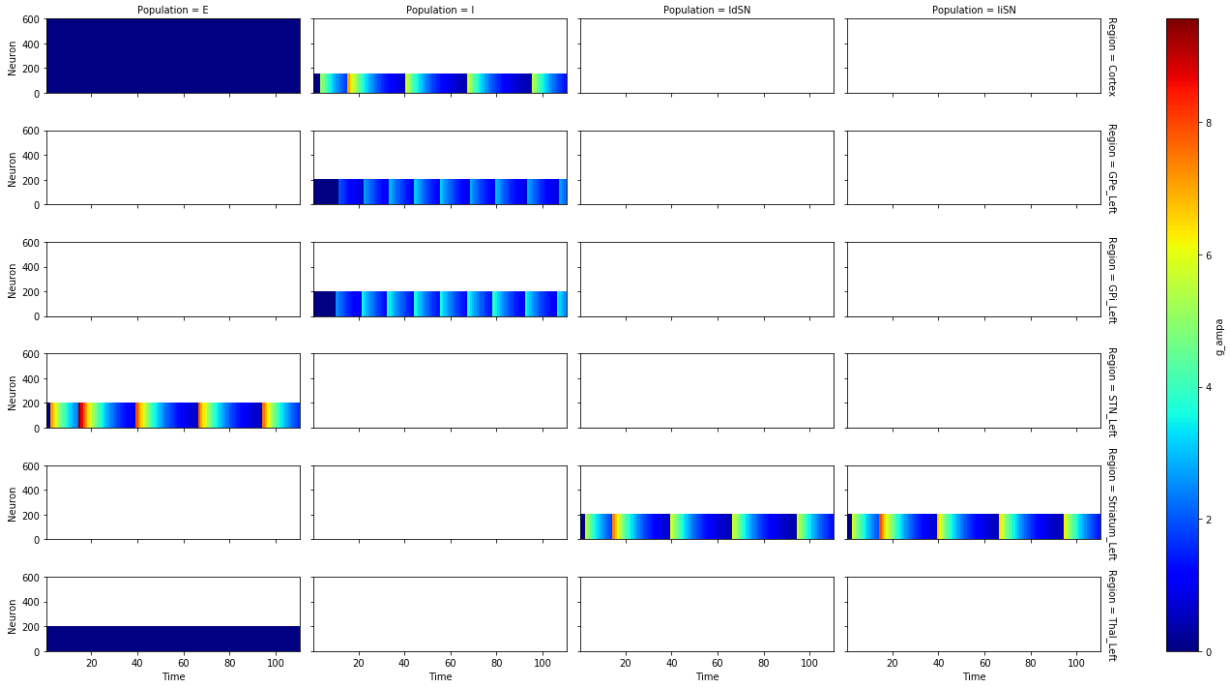

```
<Figure size 432x288 with 0 Axes>
```



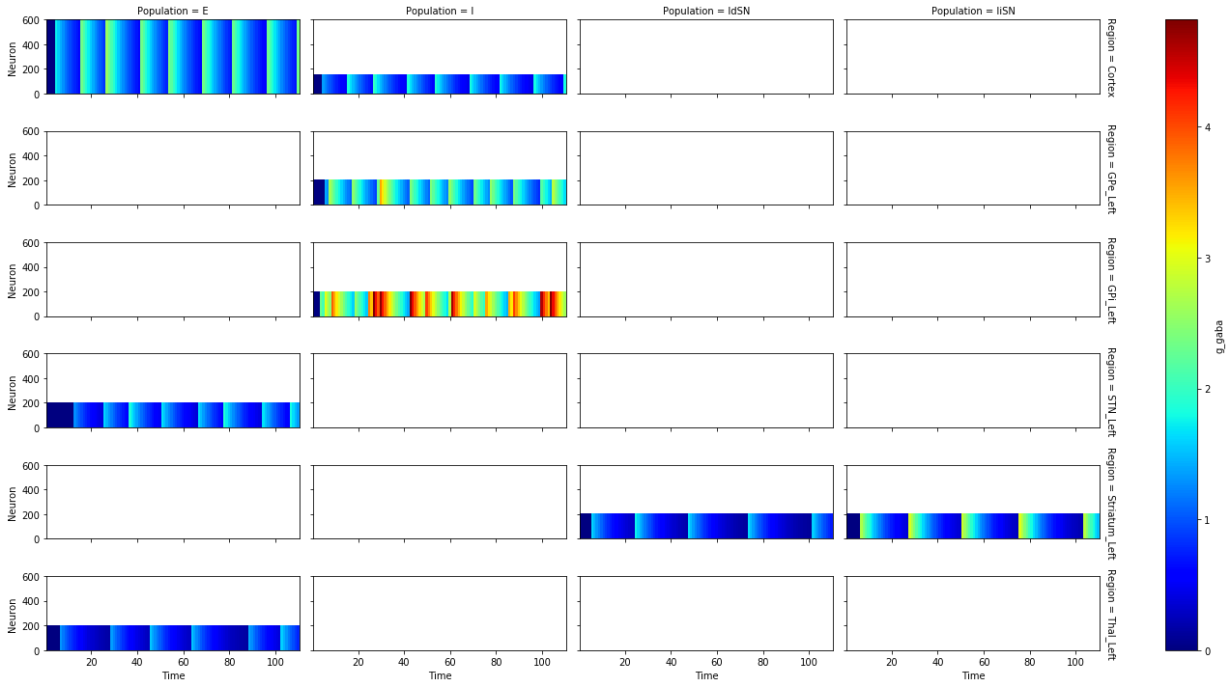

```
<Figure size 432x288 with 0 Axes>
```



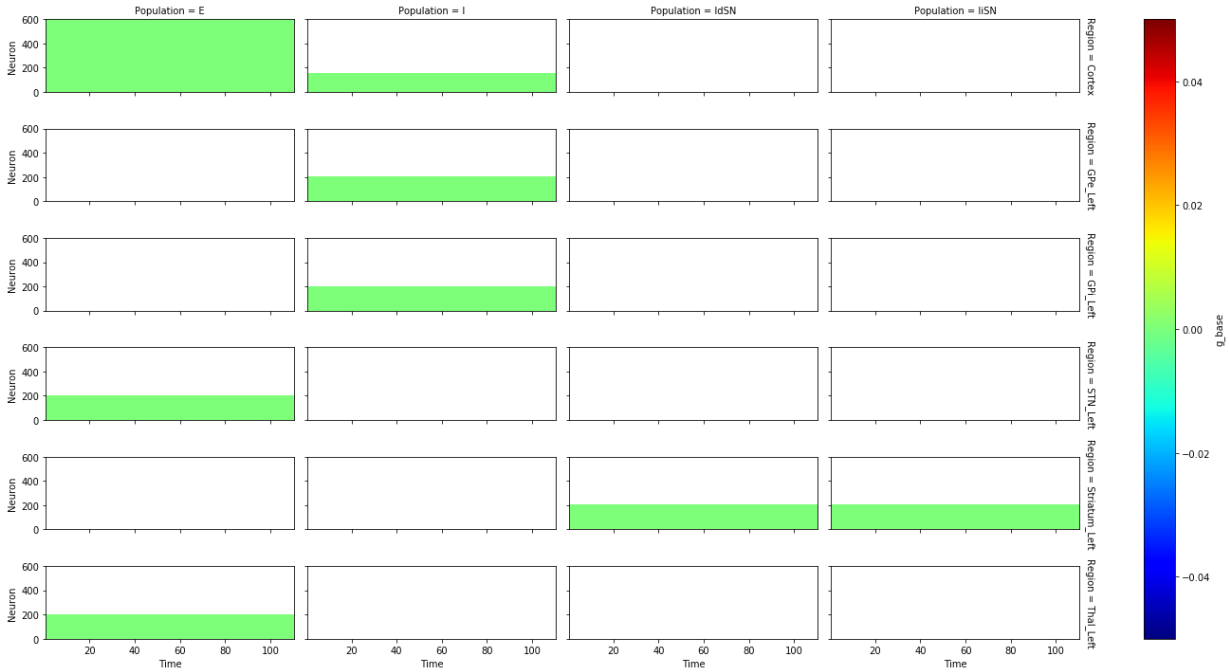
<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>



References

1 Sanz Leon P, Knock SA, Woodman MM, Domide L,
Mersmann J, McIntosh AR, Jirsa VK (2013)
The Virtual Brain: a simulator of primate brain network dynamics.
Frontiers in Neuroinformatics 7:10. doi: 10.3389/fninf.2013.00010
https://www.thevirtualbrain.org/tvb/zwei
https://github.com/the-virtual-brain

2 Ritter P, Schirner M, McIntosh AR, Jirsa VK (2013).
The Virtual Brain integrates computational modeling
and multimodal neuroimaging. Brain Connectivity 3:121–145.

3 Vitay J, Dinkelbach HÜ and Hamker FH (2015).
ANNarchy: a code generation approach to neural simulations on parallel hardware.
Frontiers in Neuroinformatics 9:19. doi:10.3389/fninf.2015.00019
For more details see https://annarchy.readthedocs.io/en/latest/

4 Baladron, J., Nambu, A., & Hamker, F. H. (2019).
The subthalamic nucleus-external globus pallidus loop biases
exploratory decisions towards known alternatives: A neuro-computational study.
European Journal of Neuroscience, 49:754–767. https://doi.org/10.1111/ejn.13666

5 Maith O, Villagrasa Escudero F, Ülo Dinkelbach H, Baladron J,
Horn, A, Irmen F, Kühn AA, Hamker FH (2020).
A computational model-based analysis of basal ganglia pathway changes
in Parkinson's disease inferred from resting-state fMRI
European Journal of Neuroscience, 00:1–18. https://doi.org/10.1111/ejn.14868

In [ ]: