NeuroChaT Documentation

Release 1.0.0

Md Nurul Islam (islammn_at_tcd_dot_ie)

Oct 02, 2018

CONTENTS

CHAPTER

ONE

NEUROCHAT.NC_BASE MODULE

This module implements two classes NAbstract and NBase those are inherited by other data classes for detailed implementation. Methods and attributes those are likely to be common in other data types in NeuroChaT are implemented in these classes.

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_base.**NAbstract**(***kwargs*)

Bases: object

Nabstract is the abstract class which includes number of attirbutes and methods commonly used by most other data types.

__init__(***kwargs*)

Instantiate the *NAbstract* class

Parameters ****kwargs** – Keyword arguments

get_comments()

Gets the comments or notes about the experiment

Parameters **None** –

Returns

Return type str

get_data_source()

Gets the source of the data

Parameters **None** –

Returns

Return type str

get_date()

Gets the recording date

Parameters **None** –

Returns

Return type str

get_duration()

Gets the duration of the experiment

Parameters **None** –

Returns

Return type str

get_experimenter()

Gets the name of the experimenter

Parameters **None** –

Returns

Return type str

get_file_version()
Gets the version of the data file

Parameters **None** –

Returns

Return type str

get_filename()
Returns the filename of the data class.

Parameters **None** –

Returns

Return type str

get_name()
Gets the name of the object.

Parameters **None** –

Returns

Return type str

get_record_info(*record_name=None*)
Gets the comments or notes about the experiment

Parameters **None** –

Returns

Return type str

get_results()
Returns the analysis results

Returns

Return type OrderedDict

get_source_format()
Gets the recording system or native data format

Parameters **None** –

Returns

Return type str

get_system()
Returns the name of the recording system or data format.

Parameters **None** –

Returns

Return type str

get_time()
Gets the time of the experiment

Parameters **None** –

Returns

Return type str

get_type()
Returns the type of data class, e.g., instance of Nabstract will return 'abstract' as the type of the class.

Parameters **None** –

Returns

Return type str

load()
Implemented in subclasses

reset_results()
Resets the results to an empty OrderedDict.

save_to_hdf5(*parent_dir*)
Implemented in subclasses

set_description(*description=''*)
Sets the general description about the data by the user.

Parameters **description** (*str*) –

Returns

Return type None

set_filename(*filename=None*)
Sets the file name of the data object

Parameters **filename** (*str*) – Name of the data file

Returns

Return type None

set_name(*name=''*)
Sets a name for the class instance.

Parameters **name** (*str*) –

Returns

Return type None

set_record_info(*new_info={}*)
Sets the recording information

Parameters **None** –

Returns Sets one of the recording information in *(name, value)* pair

Return type dict

set_system(*system=None*)
Sets the name of the recording system or the format of the data file.

Parameters **system** (*str*) – Recording system or data file format

Returns

Return type None

update_result(*new_result={}*)
Updates the results.

Parameters **description** (*str*) –

Returns

Return type OrderedDict

See also:

get_results()

class neurochat.nc_base.**NBase**(***kwargs*)

Bases: *neurochat.nc_base.NAbstract*

Derived from NAbstract class, NBase implements additional functionalities for managing multiple spike or LFP datasets.

__init__(***kwargs*)

Instantiate the *NBase* class

Parameters ****kwargs** – Keyword arguments

add_node(*node*, *node_type=None*, ***kwargs*)

Adds a new dataset, called node to the spike and LFP dataset arrays

Parameters

- **node** – Data node to be added
- **node_type** (*str*) – Type of the dataset described in each class attributes
- ****kwargs** – Keywrod arguments

Returns

Return type None

change_names(*old_names*, *new_names*, *node_type='spike'*)

Changes the names of nodes. *old_names* should have the same length as that of *new_length*

Parameters

- **old_names** (*list of str*) – List of the old names of nodes
- **new_names** (*list of str*) – List of the new names of nodes
- **node_type** – Type of the data node

Returns

Return type None

count_lfp()

Counts the number of lfp nodes

Parameters **None** –

Returns Total number of lfp nodes

Return type int

count_spike()

Counts the number of spike nodes

Parameters **None** –

Returns Total number of spike nodes

Return type int

del_lfp(*lfp*)

Deletes a node that represents LFP dataset

Parameters **lfp** – LFP node to be deleted by name or the object

Returns Index of the deleted node

Return type int

del_node(*node*)

Deletes a node that represents spike or LFP dataset

Parameters **node** – Data node to be deleted

Returns Index of deleted node

Return type int

del_spike(*spike*)
Deletes a node that represents spike dataset

Parameters **spike** – Spike node to be deleted by name or the object

Returns **i** – Index of the deleted node

Return type int

get_lfp(*names=None*)
Gets the lfp nodes by name

Parameters **names** (*list*) – List of the names of the lfp nodes to obtain

Returns List of the lfp nodes. Returns all the lfp nodes if *names* is None

Return type list

get_lfp_names()
Gets the name of all the lfp nodes

Parameters **None** –

Returns Names of the LFP nodes

Return type list

get_node(*node_names*, *node_type='spike'*)
Gets the nodes by name and dataset type

Parameters

- **node_names** (*list*) – List of the names of the data nodes to obtain
- **node_type** (*str*) – Type of the data node

Returns List of the data nodes

Return type list

get_spike(*names=None*)
Gets the spike nodes by name

Parameters **names** (*list*) – List of the names of the spike nodes to obtain

Returns List of the spike nodes. Returns all the spike nodes if *names* is None

Return type list

get_spike_names()
Gets the names of all the spike nodes

Parameters **None** –

Returns Names of the spike nodes

Return type list

set_lfp_file_names(*lfp_names*, *filenames*)
Sets the filenames for each LFP data node. *lfp_names* must be of equal length to *filenames*

Parameters

- **lfp_names** (*list of str*) – Names of the lfp nodes whose filenames are set
- **filenames** (*list of str*) – List of the filenames for each lfp node

Returns

Return type None

set_lfp_names(*names*)
Sets the names of the lfp nodes. Old names are replaced.

Parameters **names** (*list of str*) – List of new names of the lfp nodes

Returns

Return type None

set_node_file_names(*node_names*, *filenames*, *node_type='spike'*)
Sets the filenames for each data node. *node_names* must be of equal length to *filenames*

Parameters

- **node_names** (*list of str*) – Names of the nodes whose filenames are set
- **filenames** (*list of str*) – List of the filenames for each node
- **node_type** – Type of the data node

Returns

Return type None

set_spike_file_names(*spike_names*, *filenames*)
Sets the filenames for each data node. *spike_names* must be of equal length to *filenames*

Parameters

- **spike_names** (*list of str*) – Names of the spike nodes whose filenames are set
- **filenames** (*list of str*) – List of the filenames for each spike node

Returns

Return type None

set_spike_names(*names*)
Sets the names of the spike nodes. Old names are replaced

Parameters **names** (*list of str*) – List of new names of the spike nodes

Returns

Return type None

CHAPTER

TWO

NEUROCHAT.NC_CIRCULAR MODULE

This module implements CircStat Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_circular.**CircStat**(***kwargs*)

Bases: object

This class is the placeholder for the circular data and provides functionalities for calculating circular statistics.

calc_stat()

Calculates and returns all the circular statistics parameters

Parameters **None** –

Returns Returns the von Mises concentration parameter kappa

Return type dict

circ_histogram(*bins=5*)

Calculates the circular histogram of the angular coordinates

Parameters **bins** (*int*) – Angular binsize for the circular histogram

Returns

- **count** (*ndarray*) – Histogram bin count
- **ind** (*ndarray*) – Indices of the bins to which each value in input array belongs. Similar to the return values of the numpy.digitize function.
- **bins** (*ndarray*) – Histogram bins

static circ_regroup(*x*)

Circular regrouping of the angles. It unwraps the angular coordinates. For example, if the input array is x = np.ndarray([270, 340, 350, 20, 40]), the output will be y = [270, 340, 350, 380, 400] etc.

Parameters **x** (*ndarray*) – Array containg the angular coordinates

Returns **y** – Regrouped or unwrapped angular coordinates

Return type ndarray

circ_scatter(*bins=2*, *step=0.05*, *rmax=None*)

Prepares data for circular scatter plot. For each theta in a bin, the radius is increased by 'step' size capped at 'rmax'.

Parameters

- **bins** (*int*) – Angular binsize for the circular scatter
- **step** (*float*) – Stepsize to increase the radius for each count of theta
- **rmax** (*float*) – Maximum value for the radius

Returns

- **radius** (*ndarray*) – Radius for the theta values. For each new theta in a bin, the radius is increased by 'step' size.
- **theta** (*ndarray*) – Binned theta samples

circ_smooth (*filttype='b'*, *filtsize=5*)
Calculates the circular average of theta with each sample replaced by the circular mean of length 'filtsize' and weights determined by the type of filter.

Parameters

- **filttype** (*str*) – Type of smoothing filter. 'b' for Box filter, 'g' for Gaussian filter
- **filtsize** (*int*) – Length of the averaging filter

Returns **smooth_theta** – Theta values after the smoothing

Return type ndarray

get_mean_std ()
Returns the circular mean, standard deviation and resultant vector length of the data

Parameters **None** –

Returns Dictionary of mean, standard deviation and resultant vector length etc.

Return type dict

get_rayl_stat ()
Returns the Rayleigh Z statistics of the circular data

Parameters **None** –

Returns Rayleigh Z statistics

Return type dict

get_result ()
Resturns the results of the circular statistics analyses

Parameters **None** –

Returns Results of the circular statistics analyses

Return type dict

get_rho ()
Returns the radial coordinates (rho) of the circular data

Parameters **None** –

Returns Radial coordinates of the circular data

Return type ndarray

get_theta ()
Returns the angular coordinates (theta) of the circular data

Parameters **None** –

Returns Angular coordinates of the circular data

Return type ndarray

get_vonmises_stat ()
Returns the von Mises concentration parameter kappa

Parameters **None** –

Returns Returns the von Mises concentration parameter kappa

Return type dict

set_rho(*rho=None*)
Sets the radial coordinates (rho) of the circular data

Parameters **rho** (*ndarray*) – Radial coordinates of the circular data

Returns

Return type None

set_theta(*theta=None*)
Sets the angular coordinates (theta) of the circular data in degrees.

Parameters **theta** (*ndarray*) – Angular coordinates of the circular data

Returns

Return type None

CHAPTER

THREE

NEUROCHAT.NC_CLUST MODULE

This module implements NClust Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_clust.**NClust**(***kwargs*)

Bases: *neurochat.nc_base.NBase*

This class facilitates clustering-related operations. Although no clustering algorithm is implemented in this class, it can be subclassed to create such algorithms.

__init__(***kwargs*)

spike
NSpike – An object of NSpike() class or its subclass

add_spike(*spike=None*, ***kwargs*)
Adds new spike node to current NSpike() object

Parameters **spike** (NSpike) – NSPike object. If None, new object is created

Returns ‘ – A new NSpike() object

Return type obj:NSpike‘

align_wave_peak(*reach=300*, *factor=2*)
Align the waves by their peaks.

Parameters

- **reach** (*int*) – Maximum allowed time-shift in microsecond
- **factors** (*int*) – Resampling factor

Returns

Return type None

burst(*burst_thresh=5*, *ibi_thresh=50*)
Burst analysis of spik-train

Delegates to NSpike().burst()

Parameters

- **burst_thresh** (*int*) – Minimum ISI between consecutive spikes in a burst
- **ibi_thresh** (*int*) – Minimum inter-burst interval between two bursting groups of spikes

Returns

Return type None

See also:

nc_spike.NSpike()

cluster_separation(*unit_no=0*)
Quantitatively measures the separation of a specific unit from other clusters

Parameters **unit_no** (*int*) – Unit of interest. If '0', pairwise comparison of all units are returned

Returns

- **bc** (*ndarray*) – Bhattacharyya coefficient
- **dh** (*ndarray*) – Hellinger distance

cluster_similarity(*nclust=None*, *unit_1=None*, *unit_2=None*)
Quantitatively measures the similarity or distance of cluster of one unit in a spike dataset to cluster of another unit in another dataset

Parameters

- **nclust** (*Nclust*) – NClust object whose unit is under comparison
- **unit_1** (*int*) – Unit of current Nclust object
- **unit_2** (*int*) – Unit of another NClust object under comparison

Returns

- **bc** (*ndarray*) – Bhattacharyya coefficient
- **dh** (*ndarray*) – Hellinger distance

get_channel_ids()
Returns the identities of individual channels

Parameters **None** –

Returns Identities of individual channels

Return type list

get_feat(*npc=2*)
Returns the spike-waveform features.

Parameters **nc** (*int*) – Number of principle components in each channel.

Returns **feat** – Matrix of size (number_spike X number_features)

Return type ndarray

get_feat_by_unit(*unit_no=None*)
Returns the spike-waveform features for a paricular unit.

Parameters **unit_no** (*int*) – Unit of interest

Returns **feat** – Matrix of size (number_spike X number_features)

Return type ndarray

get_max_energy_chan()
Returns the maximum energy of the spike waveforms

Parameters **None** –

Returns Maximum energy of the spikes

Return type ndarray

get_max_wave_chan()
Returns the maximum of waveform peaks among the electrode groups.

Parameters **None** –

Returns

- **max_wave_val** (*ndarray*) – Maximum value of the peaks of the waveforms

- **max_wave_chan** (*ndarray*) – Channel of the electrode group where a spike waveform is strongest
- **peak_loc** (*ndarray*) – Peak location in the channel with strongest waveform

get_min_wave_chan()
Returns the maximum of waveform peaks among the electrode groups.

Parameters **None** –

Returns

- *ndarray* – Minimum value of the waveform at channels with maximum peak value
- *ndarray* – Index of minimum values

get_samples_per_spike()
Returns the number of bytes to represent each timestamp in the binary file

Parameters **None** –

Returns Number of bytes to represent timestamps

Return type int

get_sampling_rate()
Returns the sampling rate of spike waveforms

Parameters **None** –

Returns Sampling rate for spike waveforms

Return type int

get_timebase()
Returns the timebase for spike event timestamps

Parameters **None** –

Returns Timebase for spike event timestamps

Return type int

get_timestamp(*unit_no=None*)
Returns the timestamps of the spike-waveforms of specified unit

Parameters **unit_no** (*int*) – Unit whose timestamps are to be returned

Returns Timestamps of the spiking waveforms

Return type ndarray

get_total_channels()
Returns total number of electrode channels in the spike data file

Parameters **None** –

Returns Total number of electrode channels

Return type int

get_total_spikes()
Returns total number of spikes in the recording

Parameters **None** –

Returns Total number of spikes

Return type int

get_unit_list()
Returns the list of units in a spike dataset

Parameters **None** –

Returns List of units

Return type list

get_unit_spikes_count(*unit_no=None*)
Returns the total number of spikes in a specified unit

Parameters **unit_no** (*int*) – Unit whose count is returned

Returns Total number of spikes in the unit

Return type int

get_unit_tags()
Returns tags of the spiking waveforms from clustering

Parameters **None** –

Returns

Return type None

get_unit_waves(*unit_no=None*)
Returns spike waveforms of a specific unit

Parameters **unit_no** (*int*) – Unit whose waveforms are returned

Returns Spike wavefoorms in each channel of the electrode group

Return type dict

get_wave_energy()
Energy of the spike waveforms, measured as the summation of the square of samples

Parameters **None** –

Returns **energy** – Energy of spikes (num_spike X num_channels)

Return type ndarray

get_wave_min()
Returns the minimum values of the spike-waveforms.

Parameters **None** –

Returns

- **min_w** (*ndarray*) – Minimum value of the wavefforms
- **min_loc** (*ndarray*) – Index of minimum value

get_wave_pc(*npc=2*)
Returns the Principle Components of the waveforms

Parameters **npc** (*int*) – Number of principle components from waveforms of each channel

Returns **pc** – Principle components (num_waves X npc*num_channels)

Return type ndarray

get_wave_peaks()
Returns the peaks of the spike-waveforms.

Parameters **None** –

Returns

- **peak** (*ndarray*) – Spike waveform peaks in all the electrode channels (num_waves X num_channels)
- **peak_loc** (*ndarray*) – Index of peak locations

get_wave_timestamp()
Returns the temporal resolution to represent samples of spike-waves.

Parameters **None** –

Returns Number of bytes to represent timestamps

Return type int

get_waveform()
Returns the waveforms in the spike dataset

Parameters **None** –

Returns Each key represents one channel of the electrode group, each value represents the waveforms of the spikes in a matrix form (no_samples x no_spikes)

Return type dict

get_wavetime()
Returns the timestamps of the waveforms, not the spiking-event timestamp

Parameters **None** –

Returns

Return type Timestamps of the spike-waveforms

isi(*bins='auto'*, *bound=None*, *density=False*)
Calulates the ISI histogram of the spike train

Delegates to NSpike().isi()

Parameters

- **bins** (*str or int*) – Number of ISI histogram bins. If 'auto', NumPy default is used
- **bound** (*int*) – Length of the ISI histogram in msec
- **density** (*bool*) – If true, normalized historagm is calcultaed

Returns Graphical data of the analysis

Return type dict

See also:

`NSpike()`

isi_corr(***kwargs*)
Analysis of ISI autocrrelation histogram

Delegates to NSpike().isi_auto_corr()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spike.NSpike()`

load(*filename=None*, *system=None*)
Loads spike dataset from the file

Parameters **filename** (*str*) – Name of the spike file

Returns Data format or recording system

Return type system

load_spike(*names=None*)
Loads datasets of the spike nodes. Name of each node is used for obtaining the filenames.

Parameters **names** (*list of str*) – Names of the nodes to load. If None, current NSpike() object is loaded

Returns

Return type None

psth(*event_stamp*, ***kwargs*)
Calculates peri-stimulus time histogram (PSTH)

Delegates to NSpike().psth()

Parameters

- **event_stamp** (*ndarray*) – Event timestamps
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

NSpike()

remove_null_chan()
Removes the channel from the electrode group that has no spike in it

Parameters **None** –

Returns **off_chan** – Channel number that has been removed

Return type int

resample_wave(*factor=2*)
Resamples spike waveforms

Parameters **factor** (*int*) – Resampling factor

Returns

- **wave** (*dict*) – Upsampled waveforms
- *uptime ndarray* – Upsampled wave timestamps

resample_wavetime(*factor=2*)
Resamples the timestamps of spike-waveforms

Parameters **factor** (*int*) – Resampling factor

Returns

Return type Resampled timestamps

save_to_hdf5()
Stores NSpike() object to HDF5 file

Delegates to NSPike().save_to_hdf5()

Parameters **None** –

Returns

- *None*
- *Also see*
- ——
- *nc_hdf.Nhdf().save_spike()*

set_unit_tags(*new_tags=None*)
Returns tags of the spiking waveforms from clustering

Parameters **new_tags** (*ndarray*) – Array that contains the tags for spike-waveforms based on the cluster number

Returns

Return type None

wave_property()
Claulates different waveform properties for currently set unit

Delegates to NSpike().wave_property()

Parameters **None** –

Returns Graphical data of the analysis

Return type dict

See also:

`NSpike()`

CHAPTER

FOUR

NEUROCHAT.NC_CONFIG MODULE

This module implements Configuration Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_config.**Configuration**(*filename=[]*)

Bases: object

The Configuration object is the placeholder for all the settings of NeuroChaT consisting of specification of data and analysis along with the parameters for each analysis type.

It also facilitates saving these setting to a .ncfg file and retrive them from the file. The .ncfg file is a YAML-formatted file.

__init__(*filename=[]*)

filename
str – Full file name of the configuration storage (.ncfg)

format
str – Recording system or format of the data file

analysis_mode
str – Mode of analysis in NeuroChaT. Options are 'Single Unit', 'Single Session' and 'Listed Units'

mode_id
int – Numeric ID of modes in NeuroChaT, respectively 0, 1, and 2 for the three modes

graphic_format
str – File format for output graphics. Options are 'PDF' or 'Postscript'

unit_no
int – Unit number to be analyzed. Used in 'Single Unit' mode

spatial_fiel
str – Full file of the spatial dataset

spike_file
str – Full file of the spike dataset

lfp_file
str – Full file of the lfp dataset

nwb_file
str – Full file of the NWB format dataset

excel_dir
str – Full file of the Excel list of unit. Used only in 'Listed Units' mode

anayses
dict – Dictionary of analysis methods as key and their selection as boolean values

parameters
dict – It contains (key : value) pairs of parameter names and their values

get_all_modes()
Returns the analysis modes in NeuroChaT

Parameters **None** –

Returns Modes and their IDs

Return type dict

get_analysis(*name=None*)
Returns the selection of an analysis. If name is 'all', selction values for all the analyses are returned

Parameters

- **name** (*str*) – Name of the analysis. 'all' for returning values for all the analyses
- **value** (*bool*) – Boolean value to indicate analysis selection

Returns True if selected, False if not.

Return type bool

get_analysis_list()
Returns a list of analysis

Parameters **None** –

Returns

Return type list

get_analysis_mode()
Returns the mode of analysis and mode ID

Parameters **None** –

Returns

- *str* – Analysis mode set
- *int* – ID of analysis mode

get_cell_type()
Returns the type of cell set to analyse

Parameters **None** –

Returns Cell type set for analyses

Return type str

get_config_dir()
Returns the directory of configuration file

Parameters **None** –

Returns Name of the configuration file

Return type str

get_config_file()
Returns the name of the configuration file

Parameters **None** –

Returns Name of configuration file

Return type str

get_data_dir()
Returns the data directory

Parameters **None** –

Returns Data directory

Return type str

get_data_format()
Returns the data format or recording system

Parameters **None** –

Returns Data format or recording system

Return type str

get_excel_file()
Returns the filename of the Excel list

Parameters **None** –

Returns Filename of the Excel list

Return type str

get_graphic_format()
Returns output graphics file format

Parameters **None** –

Returns Export format of output graphics

Return type str

get_lfp_file()
Returns the filename of the lfp data

Parameters **None** –

Returns Filename of the lfp data

Return type str

get_nwb_file()
Returns the filename of the HDF5 data

Parameters **None** –

Returns Filename of the HDF5 data

Return type str

get_param_list()
Returns the list of all paramaeters

Parameters **None** –

Returns List of parameter names

Return type list

get_params(*name=None*)
Gets the value of parameter. If a list of parameter names are provided, a list of values are returned.

Parameters **name** (*str or list of str*) – Name of the parameter(s)

Returns Paramater value(s)

Return type params

get_params_by_analysis(*analysis=None*)
Returns the paramters and their values

Parameters **analysis** (*str*) – Name of the analysis

Returns **params** – Dictionary of parameters and their values

Return type dict

get_spatial_file()
Returns the filename of the spatial data

Parameters **None** –

Returns Filename of the spatial data

Return type str

get_spike_file()
Returns the filename of the spike data

Parameters **None** –

Returns Filename of spike data

Return type str

get_unit_no()
Returns the unit number that is already set

Parameters **None** –

Returns Unit nmber

Return type str

load_config(*filename=None*)
Imports the configuration data from a .ncfg file

Parameters **None** –

Returns **filename** – Name of the configuration file

Return type str

save_config(*filename=None*)
Exports the configuration data to .ncfg file

Parameters **filename** (*str*) – Name of the configuration file

Returns

Return type None

set_analysis(*name=None*, *value=None*)
Sets the selection of an analysis

Parameters

- **name** (*str*) – Name of the analysis
- **value** (*bool*) – Boolean value to indicate analysis selection

Returns

Return type None

set_analysis_mode(*analysis_mode=None*)
Sets the mode of analysis

Parameters **analysis_mode** (*str*) – Mode of the analysis

Returns

Return type None

set_cell_type(*cell_type=None*)
Sets the type of cell to analyse

Parameters **cell_type** (*str*) – Cell type of interest

Returns

Return type None

set_config_dir(*directory=None*)
Sets the directory of configuration file

Parameters **directory** (*str*) – Directory of configuration file

Returns

Return type None

set_config_file(*filename*)
Sets the name of the configuration file

Parameters **directory** (*str*) – Directory of configuration file

Returns

Return type None

set_data_dir(*directory=None*)
Sets the data directory

Parameters **directory** (*str*) – Data directory

Returns

Return type None

set_data_format(*file_format=None*)
Sets the format of the data or recording system

Parameters **file_format** (*str*) – Format of the data or recording system

Returns

Return type None

set_excel_file(*excel_file=None*)
Sets filename of the Excel list

Parameters **excel_file** (*str*) – Filename of the Excel list

Returns

Return type None

set_graphic_format(*graphic_format=None*)
Sets output graphics file format

Parameters **graphic_format** (*str*) – Format of output graphic export. Options are 'PDF' or 'Postscript'

Returns

Return type None

set_lfp_file(*lfp_file=None*)
Sets filename of the lfp data

Parameters **lfp_file** (*str*) – Filename of the lfp data

Returns

Return type None

set_nwb_file(*nwb_file=None*)
Sets filename of the HDF5 data

Parameters **nwb_file** (*str*) – Filename of the HDF5 data

Returns

Return type None

set_param(*name=None*, *value=None*)
Sets the value of a parameter

Parameters

- **name** (*str*) – Name of the parameter
- **value** – Value of the parameter

Returns

Return type None

set_spatial_file(*spatial_file=None*)
Sets filename of the spatial data

Parameters **spatial_file** (*str*) – Filename of the spatial data

Returns

Return type None

set_spike_file(*spike_file=None*)
Sets filename of the spike data

Parameters **spike_file** (*str*) – Filename of the spike data

Returns

Return type None

set_unit_no(*unit_no=None*)
Sets the unit no to analyse in 'Single Unit' analysis

Parameters **unit_no** (*int*) – Unit number the user is intended to analyse

Returns

Return type None

CHAPTER

FIVE

NEUROCHAT.NC_CONTROL MODULE

This module implements NeuroChaT Class for the NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_control.**NeuroChaT**(*config=<neurochat.nc_config.Configuration object>*, *data=<neurochat.nc_data.NData object>*, *parent=None*)

Bases: PyQt5.QtCore.QThread

The NeuroChaT object is the controller object in NeuroChaT and works as the backend to the NeuroChaT graphical user interface. It reads data, parameter and analysis specifications from the Configuration class and executes accordingly. It also interafces the GUI to the rest of the NeuroChaT elements.

__init__(*config=<neurochat.nc_config.Configuration object>*, *data=<neurochat.nc_data.NData object>*, *parent=None*)

ndata
NData – NData oject

config
Configuration – Configuration object

log
NLog – Central logger object

hdf
Nhdf – A Nhdf object

close_fig(*fig*)
Closes a matplotlib.fiure.Figure() object after saving it to the output PDF. A a tuple or list of such figures are provided, each of them saved and closed accordingly.

Parameters **fig** – matplotlib.fiure.Figure() or a list or tuple of them.

Returns

Return type None

close_hdf_file()
Closes the HDF5 file object.

Parameters **None** –

Returns

Return type None

close_pdf()
closes the PDF file object.

Parameters **None** –

Returns

Return type None

cluster_evaluate(*excel_file=None*)
Takes a list of unit specifications and evaluates the quality of the clustering. The results of the analysis are written back to the input Excel file.

Parameters **excel_file** (*str*) – Name of the excel file that contains data specifications

Returns

Return type None

cluster_similarity(*excel_file=None*)
Takes a list of specifications for pairwise comparison of units. The results are written back to the input Excel file.

Parameters **excel_file** (*str*) – Name of the excel file that contains unit specifications

Returns

Return type None

convert_to_nwb(*excel_file=None*)
Takes a list of datasets in Excel file and converts them into NWB file format. This method currently supports Axona and Neuralynx data formats.

Parameters **excel_file** (*str*) – Name of the excel file that contains data specifications

Returns

Return type None

execute(*name=None*)
Checks the selection of each analyses, and executes if they are selected. It also exports the plot data from individual analyses to the hdf file and figures to the graphics file that are set in the mode() method.

Parameters **name** (*str*) – Name of the unit or the unique unit ID

Returns

Return type None

exist_hdf_path(*path=''*)
Check and returns if an HDF5 file path exists

Parameters **path** (*str*) – path to HDF5 file group

Returns **exists** – True if the path exists

Return type bool

finished

get_configuration()
Returns the Configuration() object from this class.

Parameters **None** –

Returns NeuroChaT's config attribute

Return type *Configuration*

get_hdf_groups(*path=''*)
Returns the names of groups or datasets in a path

Parameters **path** (*str*) – path to HDF5 file group

Returns Names of the groups or datasets in the path

Return type list

get_neuro_data()
Returns the NData() object from this class.

Parameters **None** –

Returns NeuroChaT's ndata attribute

Return type *NData*

get_output_files()
Returns a DataFrame of output graphic files and HDF5 files after the completion of the analysis. Index are the unit IDs of the analysed units.

Parameters **None** –

Returns **op_files** – Column 1 contains the name of the output graphic files. Column 2 gives the the name of the NWB files

Return type pandas.DataFrame

get_results()
Returns the parametric results of the analyses.

Parameters **None** –

Returns **results** – Parametric results of the analysis

Return type OrderedDict

mode()
Reads the specifications and analyzes data according to the mode that is set in the Configuration file. Thsi is the principle method in NeuroChaT that sets the input and output data files and calls the execute() method for running the analyses after it sets the data and filenames to NData() object.

Parameters **None** –

Returns

Return type None

open_hdf_file(*filename=None*)
Sets the filename and opens the file object for the HDF5 file.

Parameters **filename** (*str*) – Filename of the HDF5 object

Returns

Return type None

open_pdf(*filename=None*)
Opens the PDF file object using PdfPages from matplotlib.backends.backend_pdf

Parameters **filename** (*str*) – Filename of the PDF output

Returns

Return type None

plot_data_to_hdf(*name=None*, *graph_data=None*)
Stores plot data to the HDF5 file in the '/analysis/' path

Parameters

- **name** (*str*) – Unit ID which is also the name of the group in the '/analysis/' path
- **graph_data** (*dict*) – Dictionary of data that are plotted

Returns

Return type None

reset()
Reset NeuroChaT's internal attributes and prepares it for another set of analysis or new session.

Parameters **None** –

Returns

Return type None

run()
After calling start(), the NeuroChaT thread calls this function. It verifies the input specifications and calls the mode() method.

Parameters **None** –

Returns

Return type None

set_configuration(*config*)
Sets a new Configuration() object or its subclass object.

Parameters **config** (Configuration) – Object of Configuration class or its subclass.

Returns

Return type None

set_neuro_data(*ndata*)
Sets a new NData() object or its subclass object.

Parameters **ndata** (NData) – Object of NData class or its subclass.

Returns

Return type None

update_results(*_results*)
Updates the results with new analysis results.

Parameters **_results** (*OrderedDict*) – Dictionary of the new results

Returns

Return type None

verify_units(*excel_file=None*)
Takes a list of datasets and verify the specifications of the units. The verification tool is useful for prescreening of units before the batch-mode analysis using 'Listed Units' mode of NeuroChaT

Parameters **excel_file** (*str*) – Name of the excel file that contains data specifications

Returns

Return type None

CHAPTER

SIX

NEUROCHAT.NC_DATA MODULE

This module implements NData Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_data.**NData**

Bases: object

The NData object is composed of data objects (NSpike(), NSpatial(), NLfp(), and Nhdf()) and is built upon the composite structural object pattern.

This data class is the main data element in NeuroChaT which delegates the analysis and other operations to respective objects.

__init__()

spatial
NSpatial – Spatial data object

spike
NSpike – Spike data object

lfp
Nlfp – LFP data object

hdf
NHdf – Object for manipulating HDF5 file

data_format
str – Recording system or format of the data file

angular_velocity(***kwargs*)

Analysis of unit correlation to angular head velocity (AHV) of the animal

Delegates to NSpatial().angular_velocity()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

nc_spatial.NSpatial()

border(***kwargs*)

Analysis of the firing characteristic of a unit with respect to the environmental border

Delegates to NSpatial().border()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

burst (*burst_thresh=5*, *ibi_thresh=50*)
Burst analysis of spik-train

Delegates to NSpike().burst()

Parameters

- **burst_thresh** (*int*) – Minimum ISI between consecutive spikes in a burst
- **ibi_thresh** (*int*) – Minimum inter-burst interval between two bursting groups of spikes

Returns

Return type None

See also:

`nc_spike.NSpike()`

event_trig_average (***kwargs*)
Averaging event-triggered LFP signals

Delegates to NLfp().event_trig_average()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_lfp.NLfp()`

get_data_format()
Returns the recording system or data format

Parameters **None** –

Returns

Return type str

get_lfp_file()
Gets the filename of the LFP dataset

Parameters **None** –

Returns Filename of the LFP dataset

Return type str

get_results()
Returns the parametric results of the analyses

Parameters **None** –

Returns

Return type OrderedDict

get_spatial_file()
Gets the filename of the spatial dataset

Parameters **None** –

Returns Filename of the spatial dataset

Return type str

get_spike_file()
Gets the filename of the spike dataset

Parameters **None** –

Returns Filename of the spike dataset

Return type str

get_type()
Returns the type of object. For NData, this is always *data* type

Parameters **None** –

Returns

Return type str

gradient(***kwargs*)
Analysis of gradient cell, a unit whose firing rate gradually increases as the animal traverses from the border to the cneter of the environment

Delegates to NSpatial().gradient()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

nc_spatial.NSpatial()

grid(***kwargs*)
Analysis of Grid cells characterised by formation of grid-like pattern of high activity in the firing-rate map

Delegates to NSpatial().grid()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

nc_spatial.NSpatial()

hd_rate(***kwargs*)
Analysis of the firing characteristics of a unit with respect to animal's head-direction

Delegates to NSpatial().hd_rate()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

nc_spatial.NSpatial()

hd_rate_ccw(***kwargs*)
Analysis of the firing characteristics of a unit with respect to animal's head-direction split into clockwise and counterclockwised directions

Delegates to NSpatial().hd_rate_ccw()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

hd_shift(*shift_ind=array([-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10])*, ***kwargs*)

Analysis of firing specificity of the unit with respect to animal's head direction to oberve whether it represents past direction or anicipates a future direction.

Delegates to NSpatial().hd_shift()

Parameters

- **shift_ind** (*ndarray*) – Index of spatial resolution shift for the spike event time. Shift -1 implies shift to the past by 1 spatial time resolution, and +2 implies shift to the future by 2 spatial time resoultion.
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

hd_shuffle(***kwargs*)

Shuffling analysis of the unit to see if the head-directional firing specifity is by chance or actually correlated to the head-direction of the animal

Delegates to NSpatial().hd_shuffle()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

hd_time_lapse()

Time-lapse firing proeprties of the unit with respect to the head-direction of the animal

Delegates to NSpatial().hd_time_lapse()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

interdependence(***kwargs*)

Interdependence analysis where firing rate of each variable is predicted from another variable and the distributive ratio is measured between the predicted firing rate and the caclulated firing rate.

Delegates to NSpatial().interdependence()

Parameters ****kwargs** – Keyword arguments

Returns

Return type None

See also:

`nc_spatial.NSpatial()`

isi (*bins='auto'*, *bound=None*, *density=False*)
Analysis of ISI histogram

Delegates to NSpike().isi()

Parameters

- **bins** (*str or int*) – Number of ISI histogram bins. If 'auto', NumPy default is used
- **bound** (*int*) – Length of the ISI histogram in msec
- **density** (*bool*) – If true, normalized historagm is calcultaed

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spike.NSpike()`

isi_auto_corr (*spike=None*, ***kwargs*)
Analysis of ISI autocrrelation histogram

Delegates to NSpike().isi_corr()

Parameters

- **spike** (*NSpike()*) – If specified, it calulates cross-correlation.
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spike.NSpike()`, `nc_spike.NSpike()`

load()
Loads the data from the filenames in each constituing objects, i.e, spatial, spike and LFP

Parameters **None** –

Returns

Return type None

load_lfp()
Loads LFP dataset to NLfp() object

Parameters **None** –

Returns

Return type None

load_spatial()
Loads spatial dataset from the file to NSpatial() object

Parameters **filename** (*str*) – Full file directory of the spike dataset

Returns

Return type None

load_spike()
Loads spike dataset from the file to NSpike() object

Parameters **None** –

Returns

Return type None

loc_auto_corr(***kwargs*)

Calculates the two-dimensional correlation of firing map which is the map of the firing rate of the animal with respect to its location

Delegates to NSpatial().loc_auto_corr()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

loc_rot_corr(***kwargs*)

Calculates the rotational correlation of the locational firing rate of the animal with respect to location, also called firing map

Delegates to NSpatial().loc_rot_corr()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

loc_shift(*shift_ind=array([-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10])*, ***kwargs*)

Analysis of firing specificity of the unit with respect to animal's location to oberve whether it represents past location of the animal or anicipates a future location.

Delegates to NSpatial().loc_shift()

Parameters

- **shift_ind** (*ndarray*) – Index of spatial resolution shift for the spike event time. Shift -1 implies shift to the past by 1 spatial time resolution, and +2 implies shift to the future by 2 spatial time resoultion.
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

loc_shuffle(***kwargs*)

Shuffling analysis of the unit to see if the locational firing specifity is by chance or actually correlated to the location of the animal

Delegates to NSpatial().loc_shuffle()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

loc_time_lapse(***kwargs*)
Time-lapse firing proeprties of the unit with respect to location

Delegates to NSpatial().loc_time_lapse()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

multiple_regression(***kwargs*)
Multiple-rgression analysis where firing rate for each variable, namely location, head-direction, speed, AHV, and distance from border, are used to regress the instantaneous firing rate of the unit.

Delegates to NSpatial().multiple_regression()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

phase_dist(***kwargs*)
Analysis of spike to LFP phase distribution

Delegates to NLfp().phase_dist()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_lfp.NLfp()`

place(***kwargs*)
Analysis of place cell firing characteristics

Delegates to NSpatial().place()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

plv(***kwargs*)
Calculates phase-locking value of the spike train to underlying LFP signal.

Delegates to NLfp().plv()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_lfp.NLfp()`

reset_results()
Reset the NData results to an empty OrderedDict

Parameters **None** –

Returns

Return type None

save_to_hdf5()
Stores the spatial, spike and LFP datasets to HDF5 file

Parameters **None** –

Returns

Return type None

set_data_format(*data_format=None*)
Returns the parametric results of the analyses

Parameters **data_format** (*str*) – Recording system or format of the data

Returns

Return type None

set_lfp_file(*filename*)
Sets the filename of the LFP dataset

Parameters **filename** (*str*) – Full file directory of the spike dataset

Returns

Return type None

set_lfp_name(*name*)
Sets the name of the NLfp() object

Parameters **name** (*str*) – Name of the LFP dataset

Returns

Return type None

set_spatial_file(*filename*)
Sets the filename of the spatial dataset

Parameters **filename** (*str*) – Full file directory of the spike dataset

Returns

Return type None

set_spatial_name(*name*)
Sets the name of the spatial dataset

Parameters **name** (*str*) – Name of the spatial dataset

Returns

Return type None

set_spike_file(*filename*)
Sets the filename of the spike dataset

Parameters **filename** (*str*) – Full file directory of the spike dataset

Returns

Return type None

set_spike_name(*name='C0'*)
Sets the name of the spike dataset

Parameters **name** (*str*) – Name of the spike dataset

Returns

Return type None

set_unit_no(*unit_no*)
Sets the unit number of the spike dataset to analyse

Parameters **unit_no** (*int*) – Unit or cell number to analyse

Returns

Return type None

spectrum(***kwargs*)
Analyses frequency spectrum of the LFP signal

Delegates to NLfp().spectrum()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_lfp.NLfp()`

speed(***kwargs*)
Analysis of unit correlation with running speed

Delegates to NSpatial().speed()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spatial.NSpatial()`

spike_lfp_causality(***kwargs*)
Analyses spike to underlying LFP causality

Delegates to NLfp().spike_lfp_causality()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_lfp.NLfp()`

theta_index(***kwargs*)
Calculates theta-modulation of spike-train ISI autocorrelation histogram.

Delegates to NSpike().theta_index()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spike.NSpike()`

theta_skip_index(***kwargs*)
Calculates theta-skipping of spike-train ISI autocorrelation histogram.

Delegates to NSpike().theta_skip_index()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spike.NSpike()`

update_results(*results*)
Adds new parametric results of the analyses

Parameters **results** (*OrderedDict*) – New analyses results (parametric)

Returns

Return type None

wave_property()
Analysis of wavefor characteristics of the spikes of a unit

Delegates to NSpike().wave_property()

Parameters **None** –

Returns Graphical data of the analysis

Return type dict

See also:

`nc_spike.NSpike()`

CHAPTER

SEVEN

NEUROCHAT.NC_DEFAULTS MODULE

This module implements contains the default values of input parameters as used by the Configuration Class for NeuroChaT software. The parameter are defined as {key, {key, value}} pairs where the 1st key represent the analyse for which parameter are set. The second {key, value} pairs represent the names of the parameters and their values.

@author: Md Nurul Islam; islammn at tcd dot ie

CHAPTER

EIGHT

NEUROCHAT.NC_EVENT MODULE

Created on Wed Apr 11 13:37:30 2018

@author: Raju

class neurochat.nc_event.**NEvent**(***kwargs*)

Bases: *neurochat.nc_base.NBase*

This data class is the placeholder for the dataset that contains information about external events or stimulus. Events are stored as names and tags. Each tag is a number reprsenting particular event.

add_lfp(*lfp=None*, ***kwargs*)

Adds new LFP node to current NEvent() object

Parameters **lfp** (NLfp) – NLfp object. If None, new object is created

Returns ‘ – A new NLfp() object

Return type obj:Nlfp‘

add_spike(*spike=None*, ***kwargs*)

Adds new spike node to current NEvent() object

Parameters **spike** (NSpike) – NSPike object. If None, new object is created

Returns ‘ – A new NSpike() object

Return type obj:NSpike‘

get_event_name(*event_tag=None*)

Returns name of the event from its tag

Parameters **event_tag** (*int*) –

Returns **event_name** – Name of the event

Return type str

get_event_stamp(*event=None*)

Returns timestamps for a particular event

Parameters **event** (*str or int*) – If str, represent the name of the event. If int, represents event tag

Returns **timestamp** – Timestamps of the event

Return type ndarray

get_event_train()

Returns tags for all events in the same temporal order as they are presented

Parameters **None** –

Returns Train of events as train of tags

Return type ndarray

get_tag(*event_name=None*)
Returns tag of the event from its name

Parameters **event_name** (*str*) –

Returns **event_tag** – Tag of the event

Return type int

get_timestamp()
Returns timestamps for all events

Parameters **None** –

Returns **timestamp** – Timestamps of all the events

Return type ndarray

load(*filename=None*, *system=None*)
Reads event file from the recording formats (Currently not implemented)

Parameters

- **filename** (*str*) – Full file of the event data
- **system** (*str*) – Data formta or the recording system

Returns

Return type None

load_lfp(*names=None*)
Loads datasets of the LFP nodes. Name of each node is used for obtaining the filenames

Parameters **names** (*list of str*) – Names of the nodes to load. If *all*, all LFP nodes are loaded

Returns

Return type None

load_spike(*names='all'*)
Loads datasets of the spike nodes. Name of each node is used for obtaining the filenames

Parameters **names** (*list of str*) – Names of the nodes to load. If 'all', all the spike nodes are loaded

Returns

Return type None

phase_dist(*lfp=None*, ***kwargs*)
Analysis of event to LFP phase distribution

Delegates to NLfp().phase_dist()

Parameters

- **lfp** (NLfp) – LFP object which contains the LFP data.
- ****kwargs** – Keywrod arguments

Returns Graphical data of the analysis

Return type dict

See also:

nc_lfp.NLfp()

plv(*lfp=None*, ***kwargs*)
Calculates phase-locking value of event train to underlying LFP signal.

Delegates to NLfp().plv()

Parameters

- **lfp** (NLfp) – LFP object which contains the LFP data
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_lfp.NLfp()`

psth(*event=None*, *spike=None*, ***kwargs*)
Calculates peri-stimulus time histogram (PSTH)

Parameters

- **event** – Event name or tag
- **spike** (NSpike) – NSpike object to characterize
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

set_curr_name(*name*)
Sets current event using event name

Parameters **name** (*str*) – Name of the event

Returns

Return type None

set_curr_tag(*event*)
Sets current tag of to cosider for analysis

Parameters **event** (*str or int*) – If str, represent the name of the event. If int, represents event tag

Returns

Return type None

CHAPTER

NINE

NEUROCHAT.NC_HDF MODULE

This module implements Nhdf Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_hdf.**Nhdf**(***kwargs*)

Bases: object

The Nhdf class manages the import and export of various NeuroChaT dataset to a HDF5 file dataset. It also creates and manages the nomenclature for storage paths within the file.

close()

Closes the h5py file object

Parameters **None** –

Returns

Return type None

file()

Opens the file, and returns the file object

Parameters **None** –

Returns h5py file object

Return type object

get_dataset(*group=None*, *path=''*, *name=''*)

Stores a dataset to a specific path

Parameters

- **group** (*str*) – Path of a group in HDF5 file
- **path** (*str*) – Name of the member group. This path is relative to the 'group'
- **name** (*str*) – Name of the dataset

Returns Value of the dataset

Return type ndarray or numeric objects

get_file_object()

Returns the file object that is opened using h5py

Parameters **None** –

Returns h5py file object

Return type object

static get_file_tag(*data=None*)

Resolves and returns the file tag or extension to name the group of the neural data in the HDF5 file

Parameters **data** (NSpike *or* NLfp) – Neural data objects of NeuroChaT

Returns File extention (Axona) or name (Neuralynx) of the neural datasets

Return type str

get_filename()
Returns the full file of the HDF5 dataset

Parameters **None** –

Returns

Return type str

get_groups_in_path(*path=''*)
Returns the names of groups or datasets in a path

Parameters **path** (*str*) – path to HDF5 file group

Returns Names of the groups or datasets in the path

Return type list

get_type()
Returns the type of object. For Nhdf, this is always *hdf* type

Parameters **None** –

Returns

Return type str

initialize()
Initializes the basic groups for the HDF5 file

Parameters **None** –

Returns

Return type None

resolve_analysis_path(*spike=None*, *lfp=None*)
Resolves and returns path of the dataset where analysis results will be stroed. This path is also the unique unit ID.

Parameters

- **spike** (NSpike) – Spike data object
- **lfp** (NLfp) – Lfp data object

Returns Unique unit ID resolved from spike and lfp filenames which is also the name of the path to store the data of NeuroChaT analysis

Return type str

resolve_datapath(*data=None*)
Resolves and returns path of the dataset from NeuroChaT data objects

Parameters **data** – NeuroChaT data objects

Returns Path of the NeuroChaT data

Return type str

static resolve_hdfname(*data=None*)
Resolves and returns the name of the HDF5 file from the filenames of the NeuroChaT data

Parameters **data** – One of the NeuroChaT data objects

Returns **hdf_name** – Hdf5 file name

Return type str

save_attributes(*path=None*, *attr=None*)
Stores an attribute to a group or dataset

Parameters

- **path** (*str*) – Path of a group or dataset in HDF5 file
- **attr** (*dict*) – Attribute names and values in a dictionary

Returns

Return type None

save_cluster(*clust=None*)
Stores NClust() dataset to the HDF5 file

Parameters **clust** (*NClust()*) – Cluster data object in NeuroChaT

Returns

Return type None

save_dataset(*path=None*, *name=None*, *data=None*, *create_group=True*)
Stores a dataset to a specific path

Parameters

- **path** (*str*) – Path of a group in HDF5 file
- **name** (*str*) – Name of the new dataset
- **data** (*ndarrray or list of numbers*) – Data to be stored
- **create_group** (*bool*) – If True, creates a new group if the 'path' is not in the file

Returns

Return type None

save_dict_recursive(*path=None*, *name=None*, *data=None*, *create_group=True*)
Stores a dictionary dataset to a specific path. If the dictionary is nested, it creates a group for each of the outermost keys.

Parameters

- **path** (*str*) – Path of a group in HDF5 file
- **name** (*str*) – Name of the new dataset
- **data** (*ndarrray or list of numbers*) – Data to be stored
- **create_group** (*bool*) – If True, creates a new group if the 'path' is not in the file

Returns

Return type None

save_lfp(*lfp=None*)
Stores NLfp() dataset to the HDF5 file

Parameters **lfp** (*NLfp()*) – LFP data object in NeuroChaT

Returns

Return type None

save_object(*obj=None*)
Stores a NeuroChaT dataset to the HDF5 file. It resolves the name first and then stores the data in the storage path

Parameters **obj** – One of the NeuroChaT data types

Returns

Return type None

save_spatial(*spatial=None*)
Stores NSpatial() dataset to the HDF5 file

Parameters **spatial** (NSpatial()) – Spatial data object in NeuroChaT

Returns

Return type None

save_spike(*spike=None*)
Stores NSpike() dataset to the HDF5 file

Parameters **spike** (NSpike()) – Spike data object in NeuroChaT

Returns

Return type None

set_filename(*filename=None*)
Sets the full file of the HDF5 dataset

Parameters **filename** (*str*) – Filename of the HDF5 dataset

Returns

Return type None

CHAPTER

TEN

NEUROCHAT.NC_LFP MODULE

This module implements NLfp Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_lfp.**NLfp**(***kwargs*)

Bases: *neurochat.nc_base.NBase*

This data class is the placeholder for the dataset that contains information about the neural LFP signal. It decodes data from different formats and analyses LFP signal in the recording.

add_lfp(*lfp=None*, ***kwargs*)

Adds new LFP node to current NLfp() object

Parameters **lfp** (*NLfp*) – NLfp object. If None, new object is created

Returns ‘ – A new NLfp() object

Return type obj:Nlfp‘

add_spike(*spike=None*, ***kwargs*)

Adds new spike node to current NLfp() object

Parameters **spike** (*NSpikes*) – NSPike object. If None, new object is created

Returns ‘ – A new NSpike() object

Return type obj:NSpike()‘

event_trig_average(*event_stamp=None*, ***kwargs*)

Averaging event-triggered LFP signals

Parameters

- **event_stamp** (*ndarray*) – Timestamps of the events or the spiking activities for measuring the event triggered average of the LFP signal
- ****kwargs** – Keywrod arguments

Returns Graphical data of the analysis

Return type dict

get_bytes_per_sample()

Returns the number of bytes to represent each LFP waveform sample

Parameters **None** –

Returns Number of bytes to represent each sample of the LFP waveform

Return type int

get_channel_id()

Returns the electrode channels ID

Parameters **None** –

Returns LFP channel ID

Return type str

get_file_tag()
Returns the file tag or extension for the LFP dataset. For example, Axona recordings usually have file tags like 'eeg' or 'eeg8' etc.

Parameters **None** –

Returns File tag or extension for the LFP dataset

Return type str

get_fullscale_mv()
Returns the fullscale value of the ADC in mV

Parameters **None** –

Returns Fullscale ADC value in mV

Return type int

get_samples()
Returns LFP waveform samples

Parameters **None** –

Returns Samples of the LFP signal

Return type ndarray

get_sampling_rate()
Returns the sampling rate of spike waveforms

Parameters **None** –

Returns Sampling rate for spike waveforms

Return type int

get_timestamp()
Returns the timestamps of the LFP waveform

Parameters **None** –

Returns Timestamps of the LFP signal

Return type ndarray

get_timestamp_bytes()
Returns the number of bytes to represent each timestamp in the binary file

Parameters **None** –

Returns Number of bytes to represent timestamps

Return type int

get_total_channel()
Returns total number of electrode channels in the LFP data file

Parameters **None** –

Returns Total number of electrode channels

Return type int

get_total_samples()
Returns total number of LFP samples

Parameters **None** –

Returns Total number of LFP samples

Return type ndarray

get_type()
Returns the type of object. For NLfp, this is always *lfp* type

Parameters **None** –

Returns

Return type str

load(*filename=None*, *system=None*)
Loads LFP datasets

Parameters

- **filename** (*str*) – Name of the spike datafile
- **system** (*str*) – Recording system or format of the spike data file

Returns

Return type None

See also:

`load_lfp_axona()`, `load_lfp_NLX()`, *load_lfp_NWB()*

load_lfp(*names=None*)
Loads datasets of the LFP nodes. Name of each node is used for obtaining the filenames

Parameters **names** (*list of str*) – Names of the nodes to load. If *all*, all LFP nodes are loaded

Returns

Return type None

load_lfp_Axona(*file_name*)
Decodes LFP data from Axona file format

Parameters **file_name** (*str*) – Full file directory for the lfp data

Returns

Return type None

load_lfp_NWB(*file_name*)
Decodes LFP data from NWB (HDF5) file format

Parameters **file_name** (*str*) – Full file directory for the lfp data

Returns

Return type None

load_lfp_Neuralynx(*file_name*)
Decodes LFP data from Neuralynx file format

Parameters **file_name** (*str*) – Full file directory for the lfp data

Returns

Return type None

load_spike(*names='all'*)
Loads datasets of the spike nodes. Name of each node is used for obtaining the filenames

Parameters **names** (*list of str*) – Names of the nodes to load. If None, current NSpike() object is loaded

Returns

Return type None

phase_dist (*event_stamp*, ***kwargs*)
Analysis of spike to LFP phase distribution

Parameters

- **evnet_stamp** (*ndarray*) – Timestamps of the events of spiking activities for measring the phase distribution
- ****kwargs** – Keywrod arguments

Returns Graphical data of the analysis

Return type dict

plv (*event_stamp*, ***kwargs*)
Calculates phase-locking value of the spike train to underlying LFP signal.

When 'mode'= None in the inpput kwargs, it calculates the PLV and SFC over the entire spike-train.

If 'mode'= 'bs', it bootstraps the spike-timestamps and calculates the locking values for each set of new spike timestamps.

If 'mode'= 'tr', a time-resilved phase-locking analysis is performed where the LFP signal is split into overlapped segments for each calculation.

Parameters

- **evnet_stamp** (*ndarray*) – Timestamps of the events or the spiking activities for measuring the phase locking
- ****kwargs** – Keywrod arguments

Returns Graphical data of the analysis

Return type dict

save_to_hdf5 (*file_name=None*, *system=None*)
Stores NLfp() object to HDF5 file

Parameters

- **file_name** (*str*) – Full file directory for the lfp data
- **system** (*str*) – Recoring system or data format

Returns

- *None*
- *Also see*
- ——
- *nc_hdf.Nhdf().save_lfp()*

set_channel_id (*channel_id=''*)
Sets the electrode channels ID

Parameters **channel_id** (*str*) – Channel ID for the LFP data

Returns

Return type None

set_file_tag (*file_tag*)
Sets the file tag or extension for the LFP dataset. For example, Axona recordings usually have file tags like 'eeg' or 'eeg8' etc.

Parameters **file_tag** (*str*) – File tag or extension for the LFP dataset

Returns

Return type None

spectrum(***kwargs*)
Analyses frequency spectrum of the LFP signal

Parameters ****kwargs** – Keywrod arguments

Returns Graphical data of the analysis

Return type dict

spike_lfp_causality(*spike=None*, ***kwargs*)
(Not implemented yet)

Analyses spike to underlying LFP causality

Parameters

- **spike** (NSpike) – Spike dataset which is used for the causality analysis
- ****kwargs** – Keywrod arguments

Returns Should return graphical data of the analysis. The function is not implemented yet.

Return type dict

CHAPTER

ELEVEN

NEUROCHAT.NC_PLOT MODULE

This module implements plotting functions for NeuroChaT analyses.

@author: Md Nurul Islam; islammn at tcd dot ie

neurochat.nc_plot.**angular_velocity**(*angVel_data*)
Plots the angular head velocity of the animal vs spike rate

Parameters **angVel_data** (*dict*) – Graphical data from the unit firing to angular head velocity correlation

Returns **fig1** – Scatter plot of angular velocity vs spike-rate superimposed with fitted rate

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**border**(*border_data*)
Plots the analysis results from border analysis

Parameters **border_data** (*dict*) – Graphical data from border analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Histogram of taxicab distance of active pixels
- **fig2** (*matplotlib.pyplot.Figure*) – Angular distance of pixels vs active pixel count
- **fig3** (*matplotlib.pyplot.Figure*) – Distance from border vs spike rate
- **fig4** (*matplotlib.pyplot.Figure*) – Mean distance distance from border vs firing-rate percentage

neurochat.nc_plot.**dist_rate**(*dist_data*)
Plots the firing rate vs distance from border

Parameters **dist_data** (*dict*) – Graphical data from border and gradient analysis

Returns **fig1** – Distance from border vs spike rate

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**gradient**(*gradient_data*)
Plots the results from gradient cell analysis

Parameters **border_data** (*dict*) – Graphical data from border analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Distance from border vs spike rate
- **fig2** (*matplotlib.pyplot.Figure*) – Differential firing rate vs distance from border
- **fig3** (*matplotlib.pyplot.Figure*) – Mean distance distance from border vs firing-rate percentage

neurochat.nc_plot.**grid**(*grid_data*)
Plots the results from grid analysis

Parameters **grid_data** (*dict*) – Graphical data from border analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Autocorrelation of firing rate map, superimposed with central peaks
- **fig2** (*matplotlib.pyplot.Figure*) – Rotational correlation of autocorrelation map

neurochat.nc_plot.**hd_firing**(*hd_data*)

Plots the analysis results of head directional correlation to spike-rate

Parameters **hd_data** (*dict*) – Graphical data from the unit firing to head-directional correlation

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Polar plot of head-direction during spiking-events
- **fig2** (*matplotlib.pyplot.Figure*) – Polar plot of head-direction vs spike-rate. Predicted firing rate is also plotted.

neurochat.nc_plot.**hd_rate**(*hd_data*, *ax=None*)

Plots head direction vs spike rate

Parameters

- **hd_data** (*dict*) – Graphical data from the unit firing to head-direction correlation
- **ax** (*matplotlib.pyplot.axis*) – Axis object. If specified, the figure is plotted in this axis.

Returns **ax** – Axis of the polar plot of head-direction vs spike-rate.

Return type matplotlib.pyplot.Axis

neurochat.nc_plot.**hd_rate_ccw**(*hd_data*)

Plots the analysis results of head directional correlation to spike-rate but split into counterclockwise and clockwise head-movements.

Parameters **hd_data** (*dict*) – Graphical data from the unit firing to head-direction correlation

Returns **fig1** – Polar plot of head-direction vs spike-rate in clockwise and counterclockwise head movements

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**hd_rate_time_lapse**(*hd_data*)

Plots the analysis outcome of head directional time-lapse analysis

Parameters **hd_data** (*dict*) – Graphical data from head-directional time-lapsed anlaysis

Returns **fig** – Time-lapsed head-drectional firing rate plot

Return type list of matplotlib.pyplot.Figure

neurochat.nc_plot.**hd_shuffle**(*hd_shuffle_data*)

Plots the analysis outcome of head directional shuffling analysis

Parameters **hd_shuffle_data** (*dict*) – Graphical data from head-directional shuffling anlaysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Distribution of Rayleigh Z statistics
- **fig2** (*matplotlib.pyplot.Figure*) – Distribution of Von Mises concentration parameter Kapppa

neurochat.nc_plot.**hd_spike**(*hd_data*, *ax=None*)

Plots the head-direction of the animal at the time of spiking-events in polar scatter plot.

Parameters

- **hd_data** (*dict*) – Graphical data from the unit firing to head-direction correlation
- **ax** (*matplotlib.pyplot.axis*) – Axis object. If specified, the figure is plotted in this axis.

Returns **ax** – Axis of the polar plot of head-direction during spiking events.

Return type matplotlib.pyplot.Axis

neurochat.nc_plot.**hd_spike_time_lapse**(*hd_data*)
Plots the analysis outcome of head directional time-lapse analysis

Parameters **hd_data** (*dict*) – Graphical data from head-directional time-lapsed anlaysis

Returns **fig** – Time-lapsed spike-plots

Return type list of matplotlib.pyplot.Figure

neurochat.nc_plot.**hd_time_shift**(*hd_shift_data*)
Plots the analysis outcome of head directional time-shift analysis

Parameters **hd_shift_data** (*dict*) – Graphical data from head-directional time-shift anlaysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Skaggs information content of head directional firing in shifted time of spiking events
- **fig2** (*matplotlib.pyplot.Figure*) – Peak firing rate of head directional firing in shifted time of spiking events
- **fig3** (*matplotlib.pyplot.Figure*) – Skaggs information content of head directional firing in shifted time of spiking events

neurochat.nc_plot.**isi**(*isi_data*)
Plots Interspike interval histogram and scatter plots of interval-before vs interval-after.

Parameters **isi_data** (*dict*) – Graphical data from the ISI analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Histogram of ISI
- **fig2** (*matplotlib.pyplot.Figure*) – Scatter plot of ISI-before vs ISI-after in loglog scale
- **fig3** (*matplotlib.pyplot.Figure*) – 2D histogram of the ISI-before vs ISI-after in log-log scale

neurochat.nc_plot.**isi_corr**(*isi_corr_data*)
Plots ISI correlation.

Parameters **isi_corr_data** (*dict*) – Graphical data from the ISI correlation

Returns **fig1** – ISI correlation histogram

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**lfp_spectrum**(*plot_data*)
Plots LFP spectrum analysis data

Parameters **plot_data** (*dict*) – Graphical data from the ISI correlation

Returns **fig1** – Line plot of LFP spectrum using Welch's method

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**lfp_spectrum_tr**(*plot_data*)
Plots time-resolved LFP spectrum analysis data

Parameters **plot_data** (*dict*) – Graphical data from the ISI correlation

Returns **fig1** – 3D plot of short-term FFT of the LFP signal

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**loc_auto_corr**(*locAuto_data*)
Plots the analysis outcome of locational firing rate autocorrelation

Parameters **locAuto_data** (*dict*) – Graphical data from spatial correlation of firing map

Returns **fig1** – Spatial correlation map

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**loc_firing**(*place_data*)
Plots the analysis results of locational correlation to spike-rate

Parameters **place_data** (*dict*) – Graphical data from the unit firing to head-directional correlation

Returns **fig** – Spike-plot and firing rate map in two subplots respectively

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**loc_rate**(*place_data*, *ax=None*)
Plots location vs spike rate

Parameters

- **place_data** (*dict*) – Graphical data from the unit firing to locational correlation
- **ax** (*matplotlib.pyplot.axis*) – Axis object. If specified, the figure is plotted in this axis.

Returns **ax** – Axis of the firing rate map

Return type matplotlib.pyplot.Axis

neurochat.nc_plot.**loc_rate_time_lapse**(*place_data*)
Plots the analysis outcome of locational time-lapse analysis

Parameters **place_data** (*dict*) – Graphical data from locational time-lapsed anlaysis

Returns **fig** – Time-lapsed firing-rate map

Return type list of matplotlib.pyplot.Figure

neurochat.nc_plot.**loc_shuffle**(*loc_shuffle_data*)
Plots the analysis outcome of locational shuffling analysis

Parameters **loc_shuffle_data** (*dict*) – Graphical data from head-directional shuffling anlaysis

Returns **fig1** – Distribution of Skaggs IC, sparsity and spatial coherecne in three subplots

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**loc_spike**(*place_data*, *ax=None*)
Plots the location of spiking-events (spike-plot) along with the trace of animal in the enviroment.

Parameters

- **place_data** (*dict*) – Graphical data from the correlation of unit firing to location of the animal
- **ax** (*matplotlib.pyplot.axis*) – Axis object. If specified, the figure is plotted in this axis.

Returns **ax** – Axis of the spike-plot

Return type matplotlib.pyplot.Axis

neurochat.nc_plot.**loc_spike_time_lapse**(*place_data*)
Plots the analysis outcome of locational time-lapse analysis

Parameters **place_data** (*dict*) – Graphical data from locational time-lapsed anlaysis

Returns **fig** – Time-lapsed spike-plots

Return type list of matplotlib.pyplot.Figure

neurochat.nc_plot.**loc_time_shift**(*loc_shift_data*)
Plots the analysis outcome of locational time-shift analysis

Parameters **loc_shift_data** (*dict*) – Graphical data from head-directional time-shift analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Skaggs information content of locational firing in shifted time of spiking events
- **fig2** (*matplotlib.pyplot.Figure*) – Sparsity of locational firing in shifted time of spiking events
- **fig3** (*matplotlib.pyplot.Figure*) – Coherence of locational firing in shifted time of spiking events

neurochat.nc_plot.**multiple_regression**(*mra_data*)
Plots the results of multiple regression analysis.

Parameters **mra_data** (*dict*) – Graphical data from multiple regression analysis

Returns **fig1** – Bar plot of multiple regression results

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**plv**(*plv_data*)
Plots the analysis results of Phase-locking value (PLV)

Parameters **plv_data** (*dict*) – Graphical data from the PLV analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Plot of spike-triggered average (STA)
- **fig2** (*matplotlib.pyplot.Figure*) – Plot of FFT of STA (fSTA), average power spectrum of spike-triggered LFP signals (STP), spike-field coherence and PLV in four subplots

neurochat.nc_plot.**plv_bs**(*plv_data*)
Plots the analysis results of bootstrapped Phase-locking value (PLV)

Parameters **plv_data** (*dict*) – Graphical data from the time-resolved PLV analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Plot of fSTA
- **fig2** (*matplotlib.pyplot.Figure*) – Plot of STP
- **fig3** (*matplotlib.pyplot.Figure*) – Plot of SFC

neurochat.nc_plot.**plv_tr**(*plv_data*)
Plots the analysis results of time-resolved Phase-locking value (PLV)

Parameters **plv_data** (*dict*) – Graphical data from the time-resolved PLV analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Plot of fSTA
- **fig2** (*matplotlib.pyplot.Figure*) – Plot of STP
- **fig3** (*matplotlib.pyplot.Figure*) – Plot of SFC

neurochat.nc_plot.**rot_corr**(*plot_data*)
Plots the analysis outcome of rotational correlation of spatial autocorrelation map.

Parameters **plot_data** (*dict*) – Graphical data from spatial correlation of firing map

Returns **fig1** – Rotational correlation plot

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**scatterplot_matrix**(*_data*, *names=[]*, ***kwargs*)
Plots a scatterplot matrix of subplots. Each row of "_data" is plotted against other rows, resulting in a nrows by nrows grid of subplots with the diagonal subplots labeled with "names". Additional keyword arguments are passed on to matplotlib's "plot" command. Returns the matplotlib figure object containg the subplot grid.

neurochat.nc_plot.**set_backend**(*backend*)
Sets the backend of Matplotlib

Parameters **backend** (*str*) – The new backend for Matplotlib

Returns

Return type None

See also:

`matplotlib.pyplot.switch_backend()`

neurochat.nc_plot.**speed**(*speed_data*)
Plots the speed of the animal vs spike rate

Parameters **speed_data** (*dict*) – Graphical data from the unit firing to speed correlation

Returns **fig1** – Scatter plot of speed vs spike-rate superimposed with fitted rate

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**spike_phase**(*phase_data*)
Plots the analysis results of spike-LFP phase locking

Parameters **phase_data** (*dict*) – Graphical data from the spike-LFP phase locking analysis

Returns

- **fig1** (*matplotlib.pyplot.Figure*) – Phase histogram
- **fig2** (*matplotlib.pyplot.Figure*) – Phase distribution in circular plot
- **fig3** (*matplotlib.pyplot.Figure*) – Phase-raster in one subplot, phase histogram in another

neurochat.nc_plot.**stair_plot**(*dist_data*)
Plots the stairs of mean distance vs firing-rate bands

Parameters **dist_data** (*dict*) – Graphical data from border and gradient analysis

Returns **fig1** – Mean distance distance from border vs firing-rate percentage

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**theta_cell**(*plot_data*)
Plots theta-modulated cell and theta-skipping cell analysis data

Parameters **plot_data** (*dict*) – Graphical data from the theta-modulated cell and theta skipping cell analysis

Returns **fig1** – ISI correlation histogram superimposed with fitted sinusoidal curve.

Return type matplotlib.pyplot.Figure

neurochat.nc_plot.**wave_property**(*wave_data, plots=[2, 2]*)
Plots mean +/-std of waveforms in electrode groups

Parameters

- **wave_data** (*dict*) – Graphical data from the Waveform analysis
- **plots** (*list*) – Subplot shape. [2, 2] for tetrode setup

Returns **fig1** – Matlab figure object

Return type matplotlib.pyplot.Figure

CHAPTER

TWELVE

NEUROCHAT.NC_SPATIAL MODULE

This module implements NSpatial Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_spatial.**NSpatial**(***kwargs*)

Bases: *neurochat.nc_base.NAbstract*

This data class is the placeholder for the dataset that contains information about the spatial behaviour of the animal. It decodes data from different formats and analyses the correlation of spatial information with the spiking activity of a unit.

angular_velocity(*ftimes*, ***kwargs*)

Calculates the firing rate of the unit at different binned angular head velocity.

The spike rate vs speed is fitted with a linear equation individually for the negative and positive angular velocities, and goodness of fit is measured

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

border(*ftimes*, ***kwargs*)

Analysis of the firing characteristic of a unit with respect to the environmental border

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

calc_ang_vel(*npoint=5*)

Calculates the angular head velocity of the animal from the direction data Each sample is the slope of a fitted line of five directional data centred around current sample.

Parameters **None** –

Returns

Return type None

calc_border(***kwargs*)

Identifies the border of the recording arena from the trace of the foraging of the animal in the arena

Parameters ****kwargs** – Keyword arguments

Returns

- **border_dist** (*ndarray*) – Distance of the animal from the border at each behavioural samples
- **xedges** (*ndarray*) – Pixelated edge of the x-axis
- **yedges** (*ndarray*) – Pixelated edge of the y-axis
- **dist_mat** (*ndarray*) – A matrix of distance of each pixel of the arena from the identified border

get_ang_vel()
Returns angular head velocity of the animal

Parameters **None** –

Returns Angular head velocity of the animal

Return type ndarray

get_border()
Returns animal's distance from the border

Parameters **None** –

Returns Animal's distance from the border

Return type ndarray

get_direction()
Returns head direction of the animal

Parameters **None** –

Returns Head direction of the animal

Return type ndarray

get_duration()
Returns the duration of the experiment

Parameters **None** –

Returns Duration of the experiment

Return type float

get_event_loc(*ftimes*, ***kwargs*)
Calculates location of the event from its timestamps.

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking or any other events
- ****kwargs** – Keyword arguments

Returns

- *ndarray* – Index of the events in spatial-timestamps
- *ndarray* – x-coordinates of the event location
- *ndarray* – y-ccordinates of the event location
- *ndarray* – direction of the animal at the time of the event

get_pixel_size()
Returns the pixel size of the recorded arena

Parameters **None** –

Returns Pixel size

Return type int

get_pos_x()
Returns the X-ccordinates of animal's location

Parameters **None** –

Returns X-coordinates of animal's location

Return type ndarray

get_pos_y()
Returns the Y-ccordinates of animal's location

Parameters **None** –

Returns Y-coordinates of animal's location

Return type ndarray

get_sampling_rate()
Returns the sampling rate of the spatial samples

Parameters **None** –

Returns Spatial data sampling rate

Return type int

get_speed()
Returns speed of the animal

Parameters **None** –

Returns Speed of the animal

Return type ndarray

get_time()
Returns the time of individual spatial samples

Parameters **None** –

Returns Total spatial samples

Return type int

get_timestamp()
Returns the temporal resolution of spatial samples

Parameters **None** –

Returns Temporal resolution of spatial samples

Return type int

get_total_samples()
Returns the number of spatial samples

Parameters **None** –

Returns Total spatial samples

Return type int

get_type()
Returns the type of object. For NSpatial, this is always *spatial* type

Parameters **None** –

Returns

Return type str

gradient (*ftimes*, ***kwargs*)

Analysis of gradient cell, a unit whose firing rate gradually increases as the animal traverses from the border to the cneter of the environment

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

grid (*ftimes*, ***kwargs*)

Analysis of Grid cells characterised by formation of grid-like pattern of high activity in the firing-rate map

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

hd_rate (*ftimes*, ***kwargs*)

Calculates the firing rate of the unit with respect to the head-direciton of the animal in the environment. This is calle Tuning curve.

Precited firing map from the locational firing is also calculated and distributive ratio is measured along with the Skaggs information.

Spike-plot similar to locational firing is developed but in the circular bins which shows the direction of the animal's head at each spike's occurring time.

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

hd_rate_ccw (*ftimes*, ***kwargs*)

Calculates the tuning curve but split into clock-wise vs counterclockwise head-directional movement.

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

hd_shift (*ftimes*, *shift_ind=array([-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10])*)

Analysis of firing specificity of the unit with respect to animal's head direction to oberve whether it represents past direction or anicipates a future direction.

Parameters **shift_ind** (*ndarray*) – Index of spatial resolution shift for the spike event time. Shift -1 implies shift to the past by 1 spatial time resolution, and +2 implies shift to the future by 2 spatial time resoultion.

Returns Graphical data of the analysis

Return type dict

hd_shuffle (*ftimes*, ***kwargs*)
Shuffling analysis of the unit to see if the head-directional firing specifity is by chance or actually correlated to the head-direction of the animal

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

hd_time_lapse (*ftimes*)
Calculates the tuning curve and idnetifies the location of the spiking events at certain intervals. This method is useful in observing the evolution of unit-activity as the animal traverses the environment.

Following intervals ar used: 0-1min, 0-2min, 0-4min, 0-8min, 0-16min or 0-end depending on the recording duration 0-1min, 1-2min, 2-4min, 4-8min, 8-16min or 16-end depending on the recording duration

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

interdependence (*ftimes*, ***kwargs*)
Interdependence analysis where firing rate of each variable is predicted from another variable and the distributive ratio is measured between the predicted firing rate and the caclulated firing rate.

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns

Return type None

load (*filename=None*, *system=None*)
Loads the spatial object

Parameters **None** –

Returns

Return type None

load_lfp ()
Loads the composite lfp object

Parameters **None** –

Returns

Return type None

load_spatial_Axona (*file_name*)
Loads Axona format spatial data to the NSpatial() object

Parameters **None** –

Returns

Return type None

load_spatial_NWB (*file_name*)
Loads HDF5 format spatial data to the NSpatial() object

Parameters **None** –

Returns

Return type None

load_spatial_Neuralynx (*file_name*)
Loads Neuralynx format spatial data to the NSpatial() object

Parameters **None** –

Returns

Return type None

load_spike ()
Loads the composing spike object

Parameters **None** –

Returns

Return type None

loc_auto_corr (*ftimes*, ***kwargs*)
Calculates the two-dimensional correlation of firing map which is the map of the firing rate of the animal with respect to its location

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

loc_rot_corr (*ftimes*, ***kwargs*)
Calculates the rotational correlation of the locational firing rate of the animal with respect to location, also called firing map

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

loc_shift (*ftimes*, *shift_ind=array([-10*, *-9*, *-8*, *-7*, *-6*, *-5*, *-4*, *-3*, *-2*, *-1*, *0*, *1*, *2*, *3*, *4*, *5*, *6*, *7*, *8*, *9*, *10])*, ***kwargs*)
Analysis of firing specificity of the unit with respect to animal's location to oberve whether it represents past location of the animal or anicipates a future location.

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- **shift_ind** (*ndarray*) – Index of spatial resolution shift for the spike event time. Shift -1 implies shift to the past by 1 spatial time resolution, and +2 implies shift to the future by 2 spatial time resoultion.
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

loc_shuffle (*ftimes*, ***kwargs*)
Shuffling analysis of the unit to see if the locational firing specifity is by chance or actually correlated to the location of the animal

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

loc_time_lapse (*ftimes*, ***kwargs*)
Calculates the firing rate map and idnetifies the location of the spiking events at certain intervals. This method is useful in observing the evolution of unit-activity as the animal traverses the environment.

Following intervals ar used: 0-1min, 0-2min, 0-4min, 0-8min, 0-16min or 0-end depending on the recording duration 0-1min, 1-2min, 2-4min, 4-8min, 8-16min or 16-end depending on the recording duration

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

multiple_regression (*ftimes*, ***kwargs*)
Multiple-rgression analysis where firing rate for each variable, namely location, head-direction, speed, AHV, and distance from border, are used to regress the instantaneous firing rate of the unit.

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

place (*ftimes*, ***kwargs*)
Calculates the two-dimensional firing rate of the unit with respect to the location of the animal in the environment. This is calle Firing map.

Specificity indices are measured to assess the quality of location-specific firing of the unit.

This method also plot the events of spike occurring superimposed on the trace of the animal in the arena, commonly known as Spike Plot.

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

save_to_hdf5 (*file_name=None*, *system=None*)
Save spatial dataset to HDF5 file

Parameters **None** –

Returns

Return type None

set_ang_vel(*ang_vel*)
Sets the angular head velocity (AHV) of the animal

Parameters **ang_vel** (*ndarray*) – Angular head velocity (AHV) of the animal

Returns

Return type None

set_border(*border*)
Sets the distance of the animal from the arena border

Parameters **border** (*ndarray*) – Distance of the animal from the arena border

Returns

Return type None

set_event(*event*, ***kwargs*)
Sets the NEvent() object to NSpatial().

Parameters **event** – NEvent or its childclass or NEvent() object

Returns

Return type *NEvent*()

set_event_filename(*filename=None*)
Sets the filename for the event

Parameters **filename** (*str*) – Full file of the event dataset

Returns

Return type None

set_event_name(*name=None*)
Sets the name of the event object.

Parameters **name** (*str*) – Name of the vent dataset

Returns

Return type None

set_lfp(*lfp*, ***kwargs*)
Adds the NLfp object to NSpatial object

Parameters

- **lfp** (NLfp) – NLfp object to be added to the NSpatial object. If no spike object is provided, a new NLfp() object is created.
- ****kwargs** – Keyword argumemts for creating the new NLfp instance

Returns

Return type None

set_lfp_filename(*filename=None*)
Sets file name of the lfp dataset

Parameters **name** (*str*) – Full file directory of the lfp dataset

Returns

Return type None

set_lfp_name(*name=None*)
Sets the name of the lfp dataset

Parameters **name** (*str*) – Name of the lfp dataset

Returns

Return type None

set_pixel_size(*pixel_size*)
Sets the size of pixel size by which the entire foraged arena is tessellated

Parameters **pixel_size** (*int*) – Pixel size of the foraged arena

Returns

Return type None

set_spike(*spike*, ***kwargs*)
Adds the NSpike object to NSpatial object

Parameters

- **spike** (NSpike) – NSpike object to be added to the NSpatial object. If no spike object is provided, a new NSpike() object is created.
- ****kwargs** – Keyword argumemts for creating the new NSpike instance

Returns

Return type None

set_spike_filename(*filename=None*)
Sets file name of the spike dataset

Parameters **name** (*str*) – Full file directory of the spike dataset

Returns

Return type None

set_spike_name(*name=None*)
Sets the name of the spike dataset

Parameters **name** (*str*) – Name of the spike dataset

Returns

Return type None

set_system(*system=None*)
Sets the data format or recording system.

Parameters **system** (*str*) – Data format or recording system

Returns

Return type None

static skaggs_info(*firing_rate*, *visit_time*)
Calculates the Skaggs information content of the spatial firing

Parameters

- **firing_rate** (*ndarray*) – Firing rate of the unit at each pixelated location or binned information, i.e., binned speed or head-direction
- **visit_time** (*ndarray*) – Amount of time animal spent in each pixel or bin

Returns Skaggs information content

Return type float

smooth_direction()
Smoothes the angular head direction data using a moving circular average

Parameters **None** –

Returns

Return type None

See also:

`nc_circular.CircStat()`

smooth_speed()
Smoothes the speed data using a moving-average box filter

Parameters **None** –

Returns

Return type None

static spatial_sparsity(*firing_rate*, *visit_time*)
Calculates the spatial sparsity of the spatial firing

Parameters

- **firing_rate** (*ndarray*) – Firing rate of the unit at each pixelated location
- **visit_time** (*ndarray*) – Amount of time animal spent in each pixel

Returns Spatial sparsity

Return type float

speed(*ftimes*, ***kwargs*)
Calculates the firing rate of the unit at different binned speeds.

The spike rate vs speed is fitted with a linear equation and goodness of fit is measured

Parameters

- **ftimes** (*ndarray*) – Timestamps of the spiking activity of a unit
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

CHAPTER

THIRTEEN

NEUROCHAT.NC_SPIKE MODULE

This module implements NSpike Class for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_spike.**NSpike**(***kwargs*)

Bases: *neurochat.nc_base.NBase*

This data class is the placeholder for the dataset that contains information about the neural spikes. It decodes data from different formats and analyses single units in the recording.

add_lfp(*lfp=None*, ***kwargs*)

Adds new LFP node to current NSpike() object

Parameters **lfp** (NLfp) – NLfp object. If None, new object is created

Returns ‘ – A new NLfp() object

Return type obj:Nlfp‘

add_spike(*spike=None*, ***kwargs*)

Adds new spike node to current NSpike() object

Parameters **spike** (NSpike) – NSPike object. If None, new object is created

Returns ‘ – A new NSpike() object

Return type obj:NSpike‘

burst(*burst_thresh=5*, *ibi_thresh=50*)

Analysis of bursting properties of the spiking train

Parameters

- **burst_thresh** (*int*) – Minimum ISI between consecutive spikes in a burst
- **ibi_thresh** (*int*) – Minimum inter-burst interval between two bursting groups of spikes

Returns

Return type None

get_bytes_per_sample()

Returns the number of bytes to represent each spike waveform sample

Parameters **None** –

Returns Number of bytes to represent each sample of the spike waveforms

Return type int

get_channel_ids()

Returns the identities of individual channels

Parameters **None** –

Returns Identities of individual channels

Return type list

get_fullscale_mv()
Returns the fullscale value of the ADC in mV

Parameters **None** –

Returns Fullscale ADC value in mV

Return type int

get_samples_per_spike()
Returns the number of bytes to represent each timestamp in the binary file

Parameters **None** –

Returns Number of bytes to represent timestamps

Return type int

get_sampling_rate()
Returns the sampling rate of spike waveforms

Parameters **None** –

Returns Sampling rate for spike waveforms

Return type int

get_timebase()
Returns the timebase for spike event timestamps

Parameters **None** –

Returns Timebase for spike event timestamps

Return type int

get_timestamp(*unit_no=None*)
Returns the timestamps of the spike-waveforms of spefied unit

Parameters **None** –

Returns Timestamps of the spiking waveforms

Return type ndarray

get_timestamp_bytes()
Returns the number of bytes to represent each timestamp in the binary file

Parameters **None** –

Returns Number of bytes to represent timestamps

Return type int

get_total_channels()
Returns total number of electrode channels in the spike data file

Parameters **None** –

Returns Total number of electrode channels

Return type int

get_total_spikes()
Returns total number of spikes in the recording

Parameters **None** –

Returns Total number of spikes

Return type int

get_type()
Returns the type of object. For NSpike, this is always *spike* type

Parameters **None** –

Returns

Return type str

get_unit_list()
Gets the list of the units

Parameters **None** –

Returns List of the unique tags of spiking-waveforms from clustering

Return type list

get_unit_no(*spike_name=None*)
Gets currently set unit number of the spike dataset to analyse

Parameters **None** –

Returns Unit or cell number set to analyse

Return type int

get_unit_spikes_count(*unit_no=None*)
Returns the number of spikes in a unit

Parameters **unit_no** (*int*) – Units whose spike count is returned

Returns Number of units spikes of a unit in a recording session

Return type int

get_unit_stamp()
Gets the timestamps for currently set unit to analyse

Parameters **None** –

Returns Timestamps for currently set unit

Return type list or ndarray

get_unit_tags()
Returns the unit number or tags of the clustered units

Parameters **None** –

Returns

Return type list ot ndarray

get_unit_waves(*unit_no=None*)
Returns spike waveform of a specified unit

Parameters **unit_no** (*int*) – Unit whose waveforms are to be returned

Returns Waveforms of the specified unit. If None, waveforms of currently set unit are returned

Return type OrderedDict

get_waveform()
Returns spike-waveforms

Parameters **None** –

Returns Dictionary of spiking waveforms where keys represent the channel number

Return type OrderedDict

isi (*bins='auto'*, *bound=None*, *density=False*)
Calulates the ISI histogram of the spike train

Parameters

- **bins** (*str or int*) – Number of ISI histogram bins. If 'auto', NumPy default is used
- **bound** (*int*) – Length of the ISI histogram in msec
- **density** (*bool*) – If true, normalized historagm is calcultaed

Returns Graphical data of the analysis

Return type dict

isi_corr (*spike=None*, ***kwargs*)
Calculates the correlation of ISI histogram.

Parameters

- **spike** (*NSpike()*) – If specified, it calulates cross-correlation.
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

load (*filename=None*, *system=None*)
Loads spike datasets

Parameters

- **filename** (*str*) – Name of the spike datafile
- **system** (*str*) – Recording system or format of the spike data file

Returns

Return type None

See also:

load_spike_axona(), load_spike_NLX(), *load_spike_NWB()*

load_lfp (*names='all'*)
Loads datasets of the LFP nodes. Name of each node is used for obtaining the filenames

Parameters **names** (*list of str*) – Names of the nodes to load. If *all*, all LFP nodes are loaded

Returns

Return type None

load_spike (*names=None*)
Loads datasets of the spike nodes. Name of each node is used for obtaining the filenames

Parameters **names** (*list of str*) – Names of the nodes to load. If None, current NSpike() object is loaded

Returns

Return type None

load_spike_Axona (*file_name*)
Decodes spike data from Axona file format

Parameters **file_name** (*str*) – Full file directory for the spike data

Returns

Return type None

load_spike_NWB (*file_name*)
Decodes spike data from NWB (HDF5) file format

Parameters **file_name** (*str*) – Full file directory for the spike data

Returns

Return type None

load_spike_Neuralynx (*file_name*)
Decodes spike data from Neuralynx file format

Parameters **file_name** (*str*) – Full file directory for the spike data

Returns

Return type None

phase_dist (*lfp=None*, ***kwargs*)
Analysis of spike to LFP phase distribution

Delegates to NLfp().phase_dist()

Parameters

- **lfp** (NLfp) – LFP object which contains the LFP data
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

nc_lfp.NLfp()

plv (*lfp=None*, ***kwargs*)
Calculates phase-locking value of spike train to underlying LFP signal.

Delegates to NLfp().plv()

Parameters

- **lfp** (NLfp) – LFP object which contains the LFP data
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

nc_lfp.NLfp()

psth (*event_stamp*, ***kwargs*)
Calculates peri-stimulus time histogram (PSTH)

Parameters

- **event_stamp** (*ndarray*) – Event timestamps
- ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

save_to_hdf5 (*file_name=None*, *system=None*)
Stores NSpike() object to HDF5 file

Parameters

- **file_name** (*str*) – Full file directory for the spike data

- **system** (*str*) – Recoring system or data format

Returns

- *None*
- *Also see*
- ——
- *nc_hdf.Nhdf().save_spike()*

set_unit_no (*unit_no=None*, *spike_name=None*)
Sets the unit number of the spike dataset to analyse

Parameters **unit_no** (*int*) – Unit or cell number to analyse

Returns

Return type None

set_unit_tags (*new_tags*)
Sets the number or tags of the clustered units

Parameters **new_tags** (*list or ndarray*) – Tags for each spiking wave

Returns

Return type None

spike_lfp_causality (*lfp=None*, ***kwargs*)
Analyses spike to underlying LFP causality

Delegates to NLfp().spike_lfp_causality()

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

See also:

`nc_lfp.NLfp()`

theta_index (***kwargs*)
Analysis of theta-modulation of a unit

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

theta_skip_index (***kwargs*)
Analysis of theta-skipping of a unit

Parameters ****kwargs** – Keyword arguments

Returns Graphical data of the analysis

Return type dict

wave_property ()
Claulates different waveform properties for currently set unit

Parameters **None** –

Returns Graphical data of the analysis

Return type dict

CHAPTER

FOURTEEN

NEUROCHAT.NC_UI MODULE

This module implements NeuroChaT_Ui, the main class for NeuroChaT graphical user interface. It contains other graphical and data objects and connects to the NeuroChaT class for setting configuration and analysis in NeuroChaT.

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_ui.**NeuroChaT_Ui**

Bases: PyQt5.QtWidgets.QMainWindow

accumulate_output()

Opens the UiMerge() object for the user to accumulate the selected PDF or Postscript files.

behaviour_ui()

Sets up the behaviour of NeuroChaT_ui widgets

browse()

Opens a file dialog asking the user to select spike and spatial data files. Once selected, it also set the LFP channels in the 'LFP Ch No' combo box.

cell_type_analysis(*cell_type*)

Sets the analysis checkboxes based on the type of cell selected.

cell_type_select()

Called when there is a change in the cell type selection button groups. Sets the cell type to the selected item.

clear_log()

Clears the texts in the log box.

closeEvent(*event*)

Called when NeuroChaT window is about to close. Opens a dialogue for saving the session information in NeuroChaT configuration file (.ncfg).

cluster_evaluate()

Opens a file dialog for selecting the Excel list that contains specifications for cluster evaluation and evaluates the clusters using the NeuroChaT().cluster_evaluate() method.

See also:

NeuroChaT()

compare_units()

Opens a file dialog for selecting the Excel list that contains specifications for comparing units and compares the units through NeuroChaT().cluster_similarity() method.

See also:

NeuroChaT()

convert_to_nwb()

Opens a file dialog for selecting the Excel list that contains specifications for NWB file for conversion. It then converts the files using the NeuroChaT().convert_to_nwb() method.

See also:

NeuroChaT()

data_format_select(*ind*)
Called when there is a change in the data format selection combo box. Sets the data format to the selected item.

exit_nc()
Called when 'Exit' menu item is clicked. Closes the NeuroChaT window.

export_graphic_info()
Called when 'Export graphic file info' menu is clicked. Opens a file dialogue for the selection of an Excel file, and exports the graphic file infor in the table to the file

export_results()
Called when 'Export Results' button is clicked. Opens a file dialogue for the selection of an Excel file, and exports the results in the table to the file

graphic_format_select()
Called when there is a change in the graphic format selection button groups. Sets the output graphic format to the selected item.

lfp_chan_getitems()
Returns the list of LFP files (Neuralynx) or their file extension (Axona) once the spike data is set using the 'Browse' button

load_session()
Prompts the user to select a .ncfg file and loads the settings and parameters from the file to the GUI elements.

menu_ui()
Sets up the menu items in NeuroChaT GUI

merge_output()
Opens the UiMerge() object for the user to merge the selected PDF or Postscript files.

mode_select(*ind*)
Called when there is a change in the analysis mode selection combo box. Sets the data analysis mode to the selected item.

restore_start_button()
After reactivating the start button, it displays the results in the UI table.

retranslate_ui()
Sets up the title and icon in NeuroChaT GUI

save_log()
Opens a file dialog for the user to select a text file where the current texts of the log-box are exported

save_session()
Prompts the user to select a .ncfg file and saves the current settings and parameters from the GUI elements to the file.

selectGraphicFormatUi()
Sets up the graphic format selection panel in NeuroChaT GUI

select_all()
Called when 'Select All' box is checked or unchecked. It checks or unchecks all other analyses.

select_analysis_ui()
Sets up the analysis type selection panel in NeuroChaT GUI

select_cell_type_ui()
Sets up the cell type selection panel in NeuroChaT GUI

set_lfp_chan(*value*)
Called when the selection in the 'LFP Ch No' is changed. Sets the lfp channel accordingly.

set_parameters()
Shows the UiParameters() widget once the user clicks the 'Parameters' menu item for setting the paramters.

set_unit_no(*value*)
Called when the selection in the 'Unit No' is changed. Sets the unit number accordingly.

setup_ui()
Sets up the elements of NeuroChaT_ui class

start()
Called when start button is clicked. Starts the entire backend operation in NeuroChaT

update_log(*msg*)
Updates the log-box with new message

Parameters **msg** – New log message or record

Returns

Return type None

verify_units()
Opens a file dialog for selecting the Excel list that contains specifications for verifying the units and verifies the unit using the NeuroChaT().verify_units() method.

See also:

NeuroChaT()

class neurochat.nc_ui.**ParamBoxLayout**
Bases: PyQt5.QtWidgets.QVBoxLayout

Subclass of QtWidgets.QVBoxLayout to faciliate adding new widget item to the analysis parameter selection window.

addRow(*label_1*, *widg*, *label_2*)
Adds a new row of widget using the QtWidgets.QHBoxLayout().

Parameters

- **label_1** (*str*) – Name of the parameter
- **widg** – PyQt5 widget to add
- **label_2** (*str*) – Additional description of the parameter, i.e., unit, range etc.

class neurochat.nc_ui.**UiParameters**(*parent=None*)
Bases: PyQt5.QtWidgets.QDialog

NeuroChaT user interface for setting analysis specific parameters.

ang_vel_page()
Sets the ui elements for the 'ang_vel' analysis parameters.

behaviour_ui()
Sets the behaviour of the GUI elements.

border_page()
Sets the ui elements for the 'border' analysis parameters.

burst_page()
Sets the ui elements for the 'burst' analysis parameters.

change_stack_page()
Changes the stacked widgets of parameter setting according to the analysis selected from the lis on left of the window.

gradient_page()
Sets the ui elements for the 'gradient' analysis parameters.

grid_page()
Sets the ui elements for the 'grid' analysis parameters.

hd_rate_page()
Sets the ui elements for the 'hd_rate' analysis parameters.

hd_shuffle_page()
Sets the ui elements for the 'hd_shuffle' analysis parameters.

hd_time_lapse_page()
Sets the ui elements for the 'hd_time_lapse' analysis parameters.

hd_time_shift_page()
Sets the ui elements for the 'hd_time_shift' analysis parameters.

inter_depend_page()
Sets the ui elements for the 'inter_depend' analysis parameters.

isi_corr_page()
Sets the ui elements for the 'isi_corr' analysis parameters.

isi_page()
Sets the ui elements for the 'isi' analysis parameters.

lfp_spectrum_page()
Sets the ui elements for the 'lfp_spectrum' analysis parameters.

lfp_spike_causality_page()
Sets the ui elements for the 'lfp_spike_causality' analysis parameters.

loc_rate_page()
Sets the ui elements for the 'loc_rate' analysis parameters.

loc_shuffle_page()
Sets the ui elements for the 'loc_shuffle' analysis parameters.

loc_time_lapse_page()
Sets the ui elements for the 'loc_time_lapse' analysis parameters.

loc_time_shift_page()
Sets the ui elements for the 'loc_time_shift' analysis parameters.

multiple_regresison_page()
Sets the ui elements for the 'multiple_regression' analysis parameters.

phase_lock_page()
Sets the ui elements for the 'phase_lock' analysis parameters.

set_loc_rate_filter(*filt_type*)
Sets the ui elements for the filters for locational firing rate map.

set_spat_corr_filter(*filt_type*)
Sets the ui elements for the filters for spatial autocorrelation of locational firing rate map.

setup_ui()
Sets the GUI elements for the widget.

spatial_corr_page()
Sets the ui elements for the 'spatial_corr' analysis parameters.

speed_page()
Sets the ui elements for the 'burst' analysis parameters.

spike_phase_page()
Sets the ui elements for the 'spike_phase' analysis parameters.

theta_cell_page()
Sets the ui elements for the 'theta_cell' analysis parameters.

theta_skip_cell_page()
Sets the ui elements for the 'theta_skip_cell' analysis parameters.

waveform_page()
Sets the ui elements for the 'Waveform Analysis' parameters.

class neurochat.nc_ui.**UiResults**(*parent=None*)
Bases: PyQt5.QtWidgets.QDialog

NeuroChaT user interface for displaying the analysis results and and facilitating their export.

set_data(*pd_model*)
Sets the PandasModel as the data model for the table-view.

Parameters **pd_model** (PandasModel) – PandasModel as the table-data

set_default()
Not implemented. Can be used for clearing the table and the data model underneath.

setup_ui()
Sets up the GUI elements of the widget and their behaviour. Clicking on the 'Exporr Results' button calls the NeuroChaT_Ui.export_results() method.

See also:

PandasModel()

CHAPTER

FIFTEEN

NEUROCHAT.NC_UIGETFILES MODULE

This module implements UiGetFiles Class for NeuroChaT that provides the graphical interface and functionalities for manually selecting files.

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_uigetfiles.**UiGetFiles**(*parent=None, filters=['.pdf', '.ps']*)

Bases: PyQt5.QtWidgets.QDialog

DOWN = 1

UP = -1

__init__(*parent=None, filters=['.pdf', '.ps']*)

Instantiate the UiGetFiles class.

Parameters

- **parent** – Parent widget if any
- **filters** (*list of str*) – File filters for manual selection

parent

Parent widget

filters

list of str – Approved filters

current_filter

str – Currently set filter

files

list – List of selected files

add_items()

Called if the add button is clicked. Adds selected model item to the right side selected file box if that passes the filter

Parameters **None** –

Returns

Return type None

behaviour_ui()

Sets up the behaviour of UiGetFiles class

close_dialog()

Closes the widget for file selection.

Parameters **None** –

Returns

Return type None

dir_changed(*value*)
Called if the subdirectoy combo-box in the widget is changed to update the list of new subdirectories

Parameters **value** – Newly set item number in the combo-box of subdirectories.

Returns

Return type None

done()
Sets the list of files and closes the widget.

Parameters **None** –

Returns

Return type None

filter_changed(*value*)
Called if the filter changed to update for the new selection

Parameters **value** – Currently set filter

Returns

Return type None

get_files()
Returns the list of files.

Parameters **None** –

Returns List of selected files

Return type list

hierarchy_changed()
Called if the directory hierarchy is changed

Parameters **None** –

Returns

Return type None

item_activated(*qind*)
Called if any of the model item in the list of folders and files is double-clicked

Parameters **quind** – Indix of new model item

Returns

Return type None

line_edited(*value*)
Called if the directory text box in the widget is changed to update the list of new subdirectories

Parameters **value** – Newly set text in the directory box.

Returns

Return type None

move_items(*direction='down'*)
Moves item in the item model by changing their indices.

Parameters **direction** (*str*) – Direction of moving 'down' or 'up'

Returns

Return type None

remove_items()
Removes the item which is added to the selection list. Updates the item model.

Parameters `None` –

Returns

Return type None

setup_ui()
Sets up the elements of UiGetFiles class

update_list(*directory*)
Updates the list of folders and sets the item model for the scrollable list

Parameters `directory` – New directory whose folders and files are listed

Returns

Return type None

CHAPTER

SIXTEEN

NEUROCHAT.NC_UIMERGE MODULE

This module implements UiMerge Class for NeuroChaT that provides the graphical interface and functionalities for merging and accumulating the output graphics of NeuroChaT

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_uimerge.**UiMerge**(*parent=None*)

Bases: PyQt5.QtWidgets.QDialog

This class invokes a graphical user interface where the user can upload a list of PDF or Postscript files in Excel format or can use a filepicker to manually pick the files to merge in a file or accumulate in a folder.

browse_excel_merge()

Opens a dialogue for selecting the Excel list of PDF/Postscript files and reads the file information.

Parameters **None** –

Returns

Return type None

merge_files()

Calling this method toggles the UI selection for using an Excel list or picking files manually.

Parameters **None** –

Returns

Return type None

save_in_merge()

Opens a dialogue for selecting the file or folder where the PDF/Postscript files qill be merged or accumulated.

Parameters **None** –

Returns

Return type None

select_files_merge()

Invokes the UiGetFiles class for manual selection of the PDF or Postscript files for merging or accumulating.

Parameters **None** –

Returns

Return type None

set_default()

Sets up the defaults of the GUI

Parameters **None** –

Returns

Return type None

setup_ui()
Sets up the GUI elements

Parameters **None** –

Returns

Return type None

start()
Executes the merging or accumulating operation.

Parameters **None** –

Returns

Return type None

CHAPTER

SEVENTEEN

NEUROCHAT.NC_UIUTILS MODULE

This module implements utility functions and classes for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_uiutils.**NLogBox**(*parent=None*)
Bases: PyQt5.QtWidgets.QTextEdit

Subclassed from PyQt5.QtWidgets.QTextEdit, this class creates a formatted text-editable log-box for NeuroChaT

get_text()
Returned the texts of log-box in plain text format

Parameters **None** –

Returns Plain text of log-box

Return type str

insert_log(*msg*)
Formats further the HTML 'msg' to categorally add color to the log-texts and displays it in the log-box or in any log-handler.

Parameters **msg** – Log record that is to be displayed

class neurochat.nc_uiutils.**NOut**
Bases: PyQt5.QtCore.QObject

Subclassed from PyQt5.QtCore.QObject, it implements the Qt signalling mechanism so that when a text is written using NOut() object, it emits a text to the output to an output console or file or where the emitted signal is connected to.

In NeuroChaT, the sys.stdout.write is replaced with NOut().write, which means print('some text') will print to GUI log box in GUI and to the standard output console when used in API.

emitted

write(*text*)
Emits the texts as Qt signal

Parameters **text** (*str*) – Text to be emitted

Returns

Return type None

class neurochat.nc_uiutils.**PandasModel**(*data*, *parent=None*)
Bases: PyQt5.QtCore.QAbstractTableModel

Class to populate a QT table view with a pandas dataframe and implements methods that are to be overriden

columnCount(*parent=None*)
Overrides the columnCount() methods

Parameters **parent** (*QtCore.QModelIndex*) – Specific model for item index. Usually not required.

Returns Count of column in the item model

Return type int

data(*index*, *role=0*)
Data model for the QtCore.QAbstractTableModel.

Parameters

- **index** – Pandas DataFrame index
- **role** (*Qt.ItemDataRole*) – Usually set for QtCore.Qt.DisplayRole which means the data to rendered in the form of text

Returns Returns the data in the DataFrame's 'index' location in str format

Return type str

headerData(*index*, *orientation*, *role*)
Data model for the QtCore.QAbstractTableModel.

Parameters

- **index** – Pandas DataFrame index
- **orientation** (*Qt.Orientation*) – Orientation of the data
- **role** (*Qt.ItemDataRole*) – Usually set for QtCore.Qt.DisplayRole which means the data to be rendered in the form of text

Returns

- *Data in the DataFrame().columns[index] if orientation is 'Horizontal'*
- *or DataFrame().index[index] if orientation is 'Vertical'*

rowCount(*parent=None*)
Overrides the rowCount() methods

Parameters **parent** (*QtCore.QModelIndex*) – Specific model for item index. Usually not required.

Returns Count of row in the item model

Return type int

class neurochat.nc_uiutils.**ScrollableWidget**
Bases: PyQt5.QtWidgets.QWidget

Subclassed from PyQt5.QtWidgets.QWidget, this class creates a scrollable widget.

setContents(*cont_layout*)
Sets the contents of the scrollable widget

Parameters **cont_layout** – PyQt5 layout that is the container for scrollable elements.

neurochat.nc_uiutils.**add_check_box**(*parent=None*, *position=None*, *obj_name=''*, *text=None*)
Returns a QtWidgets.QCheckBox() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object
- **text** (*str*) – Check box text

Returns Instance of QtWidgets.QCheckBox Class

Return type QtWidgets.QCheckBox

neurochat.nc_uiutils.**add_combo_box**(*parent=None*, *position=None*, *obj_name=''*)
Returns a QtWidgets.QComboBox() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object

Returns Instance of QtWidgets.QComboBox Class

Return type QtWidgets.QComboBox

neurochat.nc_uiutils.**add_double_spin_box**(*parent=None*, *position=None*, *min_val=0*, *max_val=1*, *obj_name=''*)
Returns a QtWidgets.QDoubleSpinBox() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **min_val** (*float*) – Minimum value of the spin-box
- **max_val** (*float*) – Maximum value of the spin-box
- **obj_name** (*str*) – Name of the newly created object

Returns Instance of QtWidgets.QDoubleSpinBox Class

Return type QtWidgets.QDoubleSpinBox

neurochat.nc_uiutils.**add_group_box**(*parent=None*, *position=None*, *obj_name=''*, *title=None*)
Returns a QtWidgets.QGroupBox() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object
- **title** (*str*) – Title of the group-box

Returns Instance of QtWidgets.QGroupBox Class

Return type QtWidgets.QGroupBox

neurochat.nc_uiutils.**add_label**(*parent=None*, *position=None*, *obj_name=''*, *text=None*)
Returns a QtWidgets.QLabel() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object
- **text** (*str*) – Label text

Returns Instance of QtWidgets.QLabel Class

Return type QtWidgets.QLabel

neurochat.nc_uiutils.**add_line_edit**(*parent=None*, *position=None*, *obj_name=''*, *text=None*)
Returns a QtWidgets.QLineEdit() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object
- **text** (*str*) – Line-edit text

Returns Instance of QtWidgets.QLineEdit Class

Return type QtWidgets.QLineEdit

neurochat.nc_uiutils.**add_log_box**(*obj_name*)

Returns a NLogBox() object

Parameters **obj_name** (*str*) – Name of the newly created object

Returns Instance of NLogBox Class

Return type *NLogBox*

neurochat.nc_uiutils.**add_push_button**(*parent=None*, *position=None*, *obj_name=''*, *text=None*)

Returns a QtWidgets.QPushButton() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object
- **text** (*str*) – Button text

Returns Instance of QtWidgets.QPushButton Class

Return type QtWidgets.QPushButton

neurochat.nc_uiutils.**add_radio_button**(*parent=None*, *position=None*, *obj_name=''*, *text=None*)

Returns a QtWidgets.QRadioButton() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object
- **text** (*str*) – Button text

Returns Instance of QtWidgets.QRadioButton Class

Return type QtWidgets.QRadioButton

neurochat.nc_uiutils.**add_spin_box**(*parent=None*, *position=None*, *obj_name=''*, *min_val=0*, *max_val=128*)

Returns a QtWidgets.QSpinBox() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **min_val** (*int*) – Minimum value of the spin-box
- **max_val** (*int*) – Maximum value of the spin-box
- **obj_name** (*str*) – Name of the newly created object

Returns Instance of QtWidgets.QSpinBox Class

Return type QtWidgets.QSpinBox

neurochat.nc_uiutils.**add_widget**(*parent=None*, *position=None*, *obj_name=''*)
Returns a QtWidgets.QWidget() object

Parameters

- **parent** – Parent widget
- **position** (*tuple*) – Position in the parent object
- **obj_name** (*str*) – Name of the newly created object

Returns Instance of QtWidgets.QWidget Class

Return type QtWidgets.QWidget

neurochat.nc_uiutils.**xlt_from_utf8**(*s*)

CHAPTER

EIGHTEEN

NEUROCHAT.NC_UTILS MODULE

This module implements utility functions and classes for NeuroChaT software

@author: Md Nurul Islam; islammn at tcd dot ie

class neurochat.nc_utils.**NLog**
Bases: logging.Handler

Class for handling log information (messages, errors and warnings) for NeuroChaT. It formats the incoming message in HTML and sends it to the log interface of NeuroChaT.

emit(*record*)
Formats the incoming record and

Parameters **record** – Log record to dispkay or store

Returns

Return type None

setup()
Removes all the logging handlers and sets up a new logger with HTML formatting.

Parameters **None** –

Returns

Return type None

class neurochat.nc_utils.**Singleton**
Bases: object

Creates a Singleton object created from a subclass of this class

neurochat.nc_utils.**bhatt**(*X1*, *X2*)
Calculates Bhattacharyya coefficient and Hellinger distance between two distributions

Parameters **X2** (*X1,*) – Distributions under consideration

Returns **bc, d** – Bhattacharyya coefficient and Hellinger distance

Return type float

neurochat.nc_utils.**butter_filter**(*x*, *Fs*, **args*)
Filtering function using bidirectional zero-phase shift Butterworth filter.

Parameters

- **x** (*ndarray*) – Data or signal to filter
- **Fs** (*Sampling frequency*) –
- ***kwargs** – Arguments with filter paramters

Returns Filtered signal

Return type ndarray

neurochat.nc_utils.**chop_edges**(*x*, *xlen*, *ylen*)
Chope the edges of a firing rate map if they are not visited at ll or with zero firing rate

Parameters

- **x** (*ndarray*) – Matrix of firing rate
- **xlen** (*int*) – Maximum length of the x-axis
- **ylen** (*int*) – Maximum length of the y-axis

Returns

- **low_ind** (*list of int*) – Index of low end of valid edges
- *hig_end* – Index of high end of valid edges
- **y** (*ndarray*) – Chopped firing map

neurochat.nc_utils.**corr_coeff**(*x1*, *x2*)
Correlation coefficient between two numeric series or two signals.

Parameters **x2** (*x1,*) – Input numeric array or signals

Returns Correlation coefficient of input arrays

Return type float

neurochat.nc_utils.**extrema**(*x*, *mincap=None*, *maxcap=None*)
Finds the extrema in a numeric array or a signal

Parameters

- **mincap** – Maximum value for the minima
- **maxcap** – Minimum value for the maxima

Returns

- **xmax** (*ndarray*) – Maxima values
- **imax** (*ndarray*) – Maxima indices
- **xmin** (*ndarray*) – Minima values
- **imin** (*ndarray*) – Minima indices

neurochat.nc_utils.**fft_psd**(*x*, *Fs*, *nfft=None*, *side='one'*, *ptype='psd'*)
Calculates the Fast Fourier Transform (FFT) of a signal.

Parameters

- **x** (*ndarray*) – Input signal
- **Fs** – Sampling frequency
- **nfft** (*int*) – Number of FFT points
- **side** (*str*) – 'one'-sided or 'two'-sided FFT
- **ptype** (*str*) – Calculates power-spectral density if set to 'psd'

Returns

- **x_fft** (*ndarray*) – FFT of input
- **f** (*ndarray*) – FFt frequency

neurochat.nc_utils.**find**(*X*, *n=None*, *direction='all'*)
Finds the non-zero entries of a signal or array.

Parameters

- **X** (*ndarray or list*) – Array or list of numbers whose non-zero entries need to find out

- **n** (*int*) – Number of such entries
- **direction** (*str*) – If 'all', all entries of length n are returned. If 'first', first n entries are returned. If 'last', last n entrues are returned.

Returns Indices of non-zero entries.

Return type ndarray

neurochat.nc_utils.**find2d**(*X*, *n=None*)
Finds the non-zero entries of a matrix.

Parameters

- **X** (*ndarray*) – Matrix whose non-zero entries need to find out
- **n** (*int*) – Number of such entries

Returns

- *ndarray* – x-indices of non-zero entries.
- *ndarray* – y-indices of non-zero entries.

neurochat.nc_utils.**find_chunk**(*x*)
Finds size and indeices of chunks of non-zero segments in an array

Parameters **x** (*ndarray*) – Inout array whose non-zero chunks are to be explored

Returns

- **segsize** (*ndarray*) – Lengths of non-zero chunks
- **segind** (*ndarray*) – Indices of non-zero chunks

neurochat.nc_utils.**hellinger**(*X1*, *X2*)
Calculates Hellinger distance between two distributions.

Parameters **X2** (*X1,*) – Distributions under consideration

Returns **d** – Calculated Hellinger distance

Return type float

neurochat.nc_utils.**histogram**(*x*, *bins*)
Calculates the histogram count of input array

Parameters

- **x** (*ndarray*) – Array whose histogram needs to be calculated
- **bins** – Number of histogram bins

Returns

- *ndarray* – Histogram count
- *ndarray* – Histogram bins(lowers edges)

neurochat.nc_utils.**histogram2d**(*y*, *x*, *ybins*, *xbins*)
Calculates the joint histogram count of two arrays

Parameters

- **x** (*y,*) – Arrays whose histogram needs to be calculated
- **ybins** – Number of histogram bins in y-axis
- **xbins** – Number of histogram bins in x-axis

Returns

- *ndarray* – Histogram count
- *ndarray* – Histogram bins in x-axis (lowers edges)

- *ndarray* – Histogram bins in y-axis (lowers edges)

neurochat.nc_utils.**linfit**(*X*, *Y*, *getPartial=False*)

Calculates the linear regression coefficients in least-square sense.

Parameters

- **X** (*ndarray*) – Matrix with input variables or factors (num_dim X num_obs)
- **Y** (*ndarray*) – Array of oservation data
- **getPartial** (*bool*) – Get the partial correlation coefficients if 'True'

Returns **_results** – Dictionary with results of least-square optimization of linear regression

Return type dict

neurochat.nc_utils.**nxl_write**(*file_name*, *data_frame*, *sheet_name='Sheet1'*, *startRow=0*, *startColumn=0*)

Write Pandas DataFrame to excel file. It is a wrapper for Pandas.ExcelWriter()

Parameters

- **filename** (*str*) – Name of the output file
- **data_frame** (*pandas.DataFrame*) – DataFrame to export
- **sheet_name** (*str*) – Sheet name of the Excel file where the data is written
- **startRow** (*int*) – Which row in the file the data writing should start
- **startColumn** (*int*) – Which column in the file the data writing should start

Returns

Return type None

neurochat.nc_utils.**residual_stat**(*y*, *y_fit*, *p*)

Calculates the goodness of fit and other residual statistics between observed and fitted values from a model

Parameters

- **y** (*ndarray*) – Observed data
- **y_fit** (*ndarray*) – Fitted data to a linear model
- **p** (*int*) – Model order

Returns **_results** – Dictionary of residual statistics

Return type dict

neurochat.nc_utils.**rot_2d**(*x*, *theta*)

Rotates a firing map by a specified angle

Parameters

- **x** (*ndarray*) – Matrix of firing rate map
- **theta** – Angle of rotation in theta

Returns Rotated matrix

Return type ndarray

neurochat.nc_utils.**smooth_1d**(*x*, *filttype='b'*, *filtsize=5*, ***kwargs*)

Filters a 1D array or signal.

Parameters

- **x** (*ndarray*) – Array or signal to be filtered. If matrix, each column or row is filtered individually depending on 'dir' parameter that takes either '0' for along-column and '1' for along-row filtering.
- **filttype** (*str*) – 'b' for moving average or box filter. 'g' for Gaussian filter

- **filtsize** – Box size for box filter and sigma for Gaussian filter

Returns Filtered data

Return type ndarray

neurochat.nc_utils.**smooth_2d**(*x*, *filttype='b'*, *filtsize=5*)

Filters a 2D array or signal.

Parameters

- **x** (*ndarray*) – Matrix to be filtered
- **filttype** (*str*) – 'b' for moving average or box filter. 'g' for Gaussian filter
- **filtsize** – Box size for box filter and sigma for Gaussian filter

Returns Filtered matrix

Return type smoothX

CHAPTER

NINETEEN

MODULE CONTENTS

PYTHON MODULE INDEX

n