Package 'meaRtools'

October 6, 2017

Type Package

Title Micro-Electro Array (MEA) Analysis

Version 1.0.1

Date 2017-09-19

Description
Core algorithms for MEA spike train analysis, feature extraction, statistical analysis and plotting of multiple MEA recordings with multiple genotypes and treatments, published by Gelfman et al (2017) at https://redmine.igm.cumc.columbia.edu/projects/mea/wiki/MEA_Analysis_guide.

Depends R (>= 3.2.2)

Imports lattice,tcltk,emdist,ggplot2 (>= 2.0.0), gridExtra,reshape2,plyr,gtools, Rcpp (>= 0.12.11)

LinkingTo Rcpp

License GPL(>=3)

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

RoxygenNote 6.0.1

Author Quanli Wang [aut, ctb],
Sahar Gelfman [aut, ctb, cre],
Diana Hall [aut, ctb],
Ryan Dhindsa [aut, ctb],
Matt Halvorsen [aut, ctb],
Ellese Cotterill [aut, ctb],
Tom Edinburgh [aut, ctb],
Stephen J. Eglen [aut, ctb, ths]

Maintainer Sahar Gelfman <sahar.gelfman@columbia.edu>

R topics documented:

aggregate_features *Aggregate Feature Data*

Description

Takes data from S object (MEA data structure) and makes a list of dataframes. Each dataframe corresponds to one feature, containing values for each well across each DIV of recording

Usage

```
aggregate_features(s, feat_type, parameters)
```

Arguments

s	MEA data structure
feat_type	Type of features (e.g. "spikes", "ns", "bursts")
parameters	A list of parameters, see data("parameters")

Value

A list of dataframes for a given set of features

Examples

```
data("S")
data("parameters")
s<-list()
s[[1]]<-S
spike_features = suppressWarnings( aggregate_features(s, "spike", parameters))
ns_features = suppressWarnings( aggregate_features(s, "ns", parameters) )
burst_features = suppressWarnings( aggregate_features(s, "burst", parameters) )
```

calculate_burst_features *Filter spikes and bursts in recording objects*

Description

Apply user defined filters on the spikes that were recorded and calculate spike features.

Usage

```
calculate_burst_features(s)
```

Arguments

s	A spikelist object returned from calling calculate_spike_features.

Value

Returns an 's' object containing all the spikes and bursts of all the loaded recording Robjects.

Author(s)

Diana Hall

`calculate_entropy_and_mi` *Calculate Entropy and mututal information for each treatment level*

Description

Given an MEA recording, this function computes entropy and mutual information measures for each treatment level.

Usage

```
calculate_entropy_and_mi(mea, treatments, mult_factor = 1.5, bin_size = 0.1)
```

Arguments

`mea`	The input mea spikelist object
`treatments`	The treatment levels that MI and entropy will be computed.
`mult_factor`	The multiplication factor relating to the inter quartile range used in the algorithm. It serves as a tuning parameter with a default value of 1.5.
`bin_size`	The bin size(in second) used to compute mutual information.

Value

A list object holding MI and Entropy for each treatment level.

Examples

```
library(meaRtools)

data(S)
S <- filter_nonactive_spikes(S,spikes_per_minute_min=1)

treatments <- c("treatX", "treatY")
## compute entropies and MI's
ENT.MI <- calculate_entropy_and_mi(S, treatments, mult_factor=1.5, bin_size=0.1)
data_dists <- ENT.MI[["data_dists"]]
norm_mis_per_well <- ENT.MI[["norm_mis_per_well"]]

# test for difference in mean entropy between treatmentA, treatmentB
ent <- data_dists[["ENT"]]
ent.WT <- mean(ent[[treatments[1]]])
ent.MUT <- mean(ent[[treatments[2]]])
ent.res <- wilcox.test(ent[[treatments[1]]], ent[[treatments[2]]])
cat("entropy means (WT / MUT) :", ent.WT, "/", ent.MUT, "\n")
print(ent.res)

# test for diff in mutual info btwn treatmentA, treatmentB
mi <- data_dists[["MI"]]
mi.WT <- mean(mi[[treatments[1]]])
```

```
mi.MUT <- mean(mi[[treatments[2]]])
mi.res <- wilcox.test(mi[[treatments[1]]], mi[[treatments[2]]])
cat("mutual info means (WT / MUT) :", mi.WT, "/", mi.MUT, "\n")
print(mi.res)

  plot(density(mi[[treatments[1]]]))
  lines(density(mi[[treatments[2]]]), col="red")
```

`calculate_isis` *Calculate inter spike intervals*

Description

The function calculates all the interspikes interval between all spikes of each of the channels recorded.

Usage

```
calculate_isis(s)
```

Arguments

`s`	MEA data structure

Value

Returns the MEA data structure (S object in the example) with the following new lists:

`S$isis`	list of all isis for each channel
`S$mean_isis`	mean isis for each channel
`S$sd_isis`	sd of isis for each channel

Examples

```
data("S")
S <- calculate_isis(S)
```

`calculate_network_bursts`
Compute network bursts for a list of MEA recordings.

Description

For a list of MEA recordings, ususally from the same plate at different time point,This function detects and report network burst features at the well level.

Usage

```
calculate_network_bursts(s,sigmas, min_electrodes, local_region_min_nae)
```

Arguments

s	A list of MEA recordings, typically from the same MEA plate at different time point.
sigmas	The window size used to generate network bursts.
min_electrodes	Minimum number of electrodes to call a network burst
local_region_min_nae	Indicates if an adaptive threthold method should be used.

Value

Returns an object containing summary, nb_all, nb_features, result, and nb_features_merged.

summary	brief summary
nb_all	Each well has 3 data frames with nb times, one for each smoothing window
nb_features	a list containing a data-frame for each DIV analyzed
result	for each DIV analyzed, information on the DIV, times of nb for each well and each smoothing window
nb_features_merged	data frame with nb related features averaged across DIVs

Author(s)

Quanli Wang

References

Add reference to Yi-Fan Lu's paper when it is in press.

---

calculate_network_spikes

Compute the netwrok spikes statistics from spike lists.

Description

Taken a spike list object for a set of electrodes, this function searches network spikes returns a list of all network spikes.

Usage

```
calculate_network_spikes(e, sur = 100,ns_n, ns_t)
```

Arguments

e	A spike list object for a set of electrodes.
sur	This parameter is related to the number of datapoints to be used in summmarizing mean network spikes, which will be only used for network spike diagnostics. The default value of 100 will usually be sufficient.
ns_t	global variable, time window of a network spike
ns_n	global variable, minimum number of coincident electrodes

Value

Returns a list of object, containing network spikes.

`wells`	A list of wells that network spikes were found and defined.
`ns_all`	A list of network spikes computed from the spike lists.
`well_layout`	The plate/well layout identified by the function.

References

Need to find the paper describe this method.

`calculate_spike_features` *Filter spikes and bursts in recording objects*

Description

Apply user defined filters on the spikes that were recorded and calculate spike features.

Usage

```
calculate_spike_features(r_object_files,  parameters)
```

Arguments

`r_object_files`	A list of recording Robject files
`parameters`	A list of parameters, see data("parameters")

Value

Returns an 's' object containing all the spikes and bursts of all the loaded recording Robjects.

Author(s)

Diana Hall

`calc_burst_distributions` *calculate and plot burst featues distributions*

Description

The function calculates normalized distributions of selected bursting features and plots distribution graphs of all treatments in a recording. The function also prints csv output in the /Analysis directory for downstream stats such as permutation test of treatment labels

Usage

```
calc_burst_distributions(s, min_vals = 1, xlimit = 25, bins_in_sec = 5,
feature = "non", filter_values_by_min = 0, min_values = 0, per_well = 0,
outputdir = getwd(), min_electrodes = 4, time_stamp = "DATE_TIME")
```

Arguments

s	MEA data structure
min_vals	minimum values number per electrode, electrodes with a smaller number of values than that are discarded
xlimit	max limit of values, for example: xlimit = 25 for IBI analysis means that IBIs longer than 25 seconds will not be part of distribution calculations
bins_in_sec	how many bins to cut each of the segments. For example: IBI analysis has 25 seconds as xlimit, to analyse in a 0.1 sec resolution bins_in_sec should be set to 10, for 1 sec resolution set bins_in_sec to 1
feature	what feature to analyze, options are "ibi", "isi, "nspikes_in_burst", "duration", "spikes_density_in_burst"
filter_values_by_min	should analysis disregard values with lower then filter_values_by_min number of values ? (0/1, default is 0). For example, if set to 1 for duration analysis, should analysis consider also bursts shorter than filter_values_by_min ?
min_values	disregards values with lower then filter_values_by_min , only if filter_values_by_min set to 1
per_well	should distribution analysis be performed by testing treatment differences on well level means (1) or electrode level means(0)
outputdir	output directory
min_electrodes	minimum electrodes for an active well
time_stamp	time stamp for the output files

Details

Plot distributions calculates normalized distributions of bursting features. 'Normalized distribution' are a way to handle biases caused by noisy electrodes/wells. The function will calculate a normalized histogram (values 0-1) of each feature for each electrode. Next, it will average histogram values either per well and then average all wells per treatment, or directly per treatment. All comparisons between treatments will be then made by plotting the normalized histograms of each treatment and running a K-S test between them.

Note

Output is a made of: 1) Plots of all selected burst features distributions. 2) CSV files ending with _distributions.csv that harbor all electrodes per treatment for all the recordings loaded in the meaRtools pipeline for a specific MEA plate

Author(s)

Sahar Gelfman

Examples

```
# Load exapmple of recording Robject (MEA data structure)
data("S")
feature="ibi";

   calc_burst_distributions(S, min_vals = 15, xlimit = 20, bins_in_sec = 5,
           feature = feature, per_well = 0, outputdir = "/Analysis")
```

`calc_burst_summary` *Calculate average and standard deviation of the bursting features.*

Description

The function calculates a summary of all the bursting features and returns a data.frame with those values.

Usage

```
calc_burst_summary(s, bursty_threshold = 1)
```

Arguments

`s`	MEA data structure
`bursty_threshold`	min number of bursts/minute to count as a bursty unit.

Value

A data frame with the following columns:

`channels`	electrode name
`spikes`	#spikes
`mean_freq`	firing rate (Hz)
`nbursts`	#bursts detected
`bursts_per_sec`	#bursts/second.matrix(nrow=0,ncol=1)
`bursts_per_min`	#bursts/min
`bursty`	is bursts_per_min >bursty_threshold (defaults to 1 burst/min)
`mean_dur`	mean burst duration
`sd_dur`	sd
`mean_spikes`	mean #spikes in a burst
`sd_spikes`	sd
`per_spikes_in_burst`	% of spikes in a burst
`per_spikes_out_burst`	% of spikes not in a burst
`mean_si`	mean Surprise Index (only for poisson .surprise measure)

mean_isis	mean ISI within a burst
sd_mean_isis	sd
mean_ibis	mean IBI
sd_ibis	sd
cv_ibis	Coefficient of variation of IBI (= mean_ibi/sd_ibi)

Examples

```
# Load exapmple of recording Robject (MEA data structure)
data("S")
S$bs<-calc_burst_summary(S)
```

---

compute_mean_firingrate_by_well
mean.firingrate.by.well

Description

Creates a data frame with columns for well firing rate, mean electrode firing rate, well name and DIV. See details for computations.

Usage

```
compute_mean_firingrate_by_well(s)
```

Arguments

s	Well firing rate= total spikes per well/recording time.

Details

Well firing rate= total spikes per well/recording time. Electrode level firing rate= average across all electrodes in a well(total spikes on electrode/recording time)

Examples

```
data("S")

res<-compute_mean_firingrate_by_well(S)
res[1:4,]
```

compute_mean_sttc_by_well

Compute the mean STTC averaged across all pairwise electrodes in well

Description

Compute the mean STTC averaged across all pairwise electrodes in well

Usage

```
compute_mean_sttc_by_well(s, dt = 0.05, beg = NULL, end = NULL)
```

Arguments

s	structure storing the well information
dt	Time window for STTC (default = 0.05 secons)
beg	Start time in seconds (defaults to start of recording)
end	End time in seconds (defaults to end of recording)

Details

For each pair of electrodes, we calculate the STTC. We then take the mean of these pairs, excluding autocorrelations. If a well has one (or no) electrodes, the value returned for that well is NULL.

Warning: taking the mean over a well is useful only if you do not suspect distance-dependent correlations in your firing. (For activity like retinal waves, we find that correlations are strongly dependent on the distance separating electrodes.)

Value

A vector giving the mean of all pairwise STTCs on each well.

Author(s)

Stephen Eglen

count_ns

Count number of spikes within evenly spaced time intervals(bins) from input spike trains.

Description

Given a list of spike trains, this function creates evenly spaced bins and returns number of spikes from all spike trains for each bins.

Usage

```
count_ns(spikes, beg, end, wid, nbins)
```

Arguments

spikes	The input list of spike trains.
beg	Start time of recording in seconds.
end	End time of recording in seconds.
wid	Bin width in seconds.
nbins	Number of evenly spaced bins for given time interval.

Value

Return a vector of counts of spikes from all spike trains for user-defined, evenly spaced bins.

Author(s)

Stephen Eglen

---

dist_perm *Burst distribution permutations*

Description

Perform two statistical tests to quantify difference between two burst probability distributions using burst probability distribution data. Performed test are the Maximum Distance between cumulative distributions and Earth Movers Distance between the original probability distributions.

Usage

```
dist_perm(datafile,np,type,kotype)
```

Arguments

datafile	A _distributions.csv input file. Format as the output of calc_burst_distributions
np	Number of permutations to perform
type	Name of first genotype
kotype	Name of second genotype

Value

A list containing results of two statistical tests for the input probability distributions data.

data.EMD	Original value of EMD distance
data.EMD	Original value of maximum distance
perm.EMD	A permuted p.value of the EMD distance
perm_p	A permuted p.value of the maximum distance
outp	Maximum distances between genotypes for all permutations performed
out_emd	Maximum Earth Movers Distance between genotypes for all permutations performed
data.wt	Cumulative probabilities of the first genotype

`data.ko` Cumulative probabilities of the second genotype

`data.wt.Original`
Probabilities distribution of the first genotype

`data.ko.Original`
Probabilities distribution of the second genotype

References

See https://redmine.igm.cumc.columbia.edu/projects/mea/wiki for further details

Examples

```
result <- dist_perm(distributionFilePath,10000,"WT","KO")
```

`filter_nonactive_spikes` *Filter nonactive spikes from recordings*

Description

Given an input MEA recording, this function removes spike trains that are deemed as nonactive based on the number of spikes per minutes.

Usage

```
filter_nonactive_spikes(mea, spikes_per_minute_min = 1)
```

Arguments

`mea` The input MEA spikelist object.

`spikes_per_minute_min`
Minimum number of spikes per minute for a spike train to be considered as active.

Value

An MEA spikelist object with nonactive spike trains removed.

filter_wells *Filter wells*

Description

Filter out wells for which the number of active electrodes is less than 4, at least 70 percent of the time

Usage

```
filter_wells(unfiltered_df, nae,min_electrodes = 4,
well_max_div_inactive_ratio = 0.5)
```

Arguments

unfiltered_df	Dataframe generated by the spike_features() function
nae	A dataframe containing the number of active electrodes for the recording
min_electrodes	Minimum number of active electrode to consider a well for analysis
well_max_div_inactive_ratio	The DIV inactive/active well ratio below which a well will be considered active for a set of DIVs

Value

A dataframe identical in format to the input, except that wells that do not meet the filtering criteria are removed.

Examples

```
data("S")
data("parameters")
s<-list(); s[[1]]<-S
spike_features<-aggregate_features(s, feat_type="spike", parameters )
nae = spike_features$nae
  filtered.spike.features = lapply(spike_features, function(x) filter_wells(x, nae))
```

frate_counts *Estimate population firing rate using fixed-width time bins.*

Description

Estimate the population firing rate, averaging over all spikes.

Usage

```
frate_counts(spikes, beg, end, wid, nbins)
```

Arguments

spikes	List of simultaneously recorded spike trains
beg	Start of the recording, in seconds.
end	The start time of the last bin, in seconds.
wid	The duration of each bin
nbins	The number of bins to generate.

Details

We compute the array-wide average activity for a list of spike trains. The duration of the recording is given in seconds by BEG and END. Time is divided up into NBINS bins, each of duration WID. Each spike is then placed in the appropriate bin and then we return the average count in each bin.

Value

The population firing rate (in Hz) for each bin.

Author(s)

Stephen Eglen

---

generate_raster_plot *generate_raster_plot*

Description

Creates a pdf raster plot of selected user selected well from an 's' object. Options include verticle lines showing network spike times, vertical bars showing bursts as well as # showing count of spikes in burst and network spikes.

Usage

```
generate_raster_plot(r_object_file = NULL, outputdir = NULL, well_for_raster
                 = NULL, interval_for_raster = NULL, show_bursts = F,
                 show_burst_number = F, show_networkspikes = F,
                 show_ns_number = F,  show_nb=F, window_size=NULL)
```

Arguments

r_object_file	Default value is NULL, in which case tcltk pop-up file chooser will prompt user to select an 's' object. Otherwise, provide a full path to to a .RData 's' object that contains burst and network data.
outputdir	A directory (character string in quotes) where pdf is to be saved. Default is NULL, in which case the plot will be saved in the directory r_object_file location.
well_for_raster	A well name, character string, from plate. e.g. well_for_raster="A3". Default is NULL, in which case first well in plate will appear in plot.

`interval_for_raster`
: A vector of min and max time (s) for raster marks. e.g. interval_for_raster=c(30,60) Default is NULL, in which case the whole recording interval will be used.

`show_bursts`
: A boolean value sets whether bursts are indicated by red horizontal line (TRUE/FALSE) e.g. show_bursts=FALSE Default=FALSE

`show_burst_number`
: A boolean value sets whether # spikes/bursts are indicated (TRUE/FALSE). show_bursts must be set to true in order that show_burst_number=T e.g. show_burst_number=FALSE Default=FALSE

`show_networkspikes`
: A boolean value sets whether network spikes are indicated by green vertical line (TRUE/FALSE) e.g. show_networkspikes=FALSE Default=FALSE

`show_ns_number`
: A boolean value sets whether # electrodes in network spikes are indicated (TRUE/FALSE) e.g. show_ns_number=FALSE Default=FALSE

`show_nb`
: A boolean value sets whether network bursts should be indicated in raster by orange horizontal lines (TRUE/FALSE) e.g. show_ns_number=FALSE Default=FALSE

`window_size`
: A numeric value indicating which of the three smoothing sizes available in the R-object should be used in network burst identification e.g. show_ns_number=10Default=NULL

Value

A pdf raster plot will be displayed in system viewer.

Author(s)

Diana Hall

Examples

```
generate_raster_plot(r_object_file=NULL,
                     well_for_raster=NULL,
                     interval_for_raster=NULL,
                     show_bursts=F,
                     show_burst_number=F,
                     show_networkspikes=F,
                     show_ns_number=F,
                     show_nb=F,
                     window_size=NULL )
```

---

get_burst_info *get burst feature information*

Description

The function returns a list of values of a burst feature for a desired channel

Usage

```
get_burst_info(allb, index)
```

Arguments

allb	The bursting features matrix of a channel (located in recording object - S object in example: S$allb[[channel number]]
index	Name of the requested burting feature. Can be "beg","end", "ibi","len", "durn", "mean_isis" or "si"".

Value

List of values of the requested feature (index) for the desired channel.

Examples

```
data("S")
S$allb[[1]]
```

---

get_data *get_data*

Description

pop up file chooser with caption. Also, sets directory of analysis output.

Usage

```
get_data(caption = "")
```

Arguments

caption	text to display in pop-up file chooser to prompt user to select appropriate file. Default is no caption.

Value

Creates 2 directories:

'Analysis' directory in the parent directory of user selected file.

'R_objects' a subdirectory of 'Analysis'

Examples

```
get_data(caption="Please select a spike-list file for analysis")
```

get_experimental_log_file *get_experimental_log_file*

Description

Extract data from experimental log file: a csv file with columns for well, treatment, dose, size and units.

Usage

```
get_experimental_log_file(file, master_chem_file = master_chem_file)
```

Arguments

file	spike-list csv file, one of the possible plate recording file formats available from Axion. Format: one spike and corresponding electrode name per row. See Axion biosystems website for details.
master_chem_file	A csv file containing the following columns: "Project", "Experiment.Date", "Plate.SN", "DIV", "Well", "Treatment", "Size", "Dose", and "Units". Empty wells must still be represented in file. If column is irrelavent to a given data set, then 'NA' or blank is sufficient. "Project" columm must match the first character string preceeding "_" in spike-list file name. e.g. exampleRecording_1012016_plate1_DIV1_spike_list.csv". Similarly, "Experiment.Data" and "Plate.SN" must match second and third character strings as separated by "_" in spike-list file name. "DIV" column does not need to be matched.

Value

list containing character vector of experimental log information.

well	well name e.g. "A4""
treatment	treatment on well e.g. 'WT'
size	size information of chemical treatment
dose	dose information for treatment
units	units of dosage e.g. uL/g

References

See http://www.axionbiosystems.com/products/software/ for details on spike-list csv file format

Examples

```
master_chem_file<-paste0( system.file(package = "meaRtools"),"/data",
        "/exampleRecording_1012016_plate1_expLog.csv" )

spike.list.file<-paste0( system.file(package = "meaRtools"),"/data",
        "/exampleRecording_1012016_plate1_DIV1_spike_list.csv" )
```

```
plate.data<-getxperimental.log.file( file=spike.list.file, master_chem_file = master_chem_file )
```

`get_file_basename` *get_file_basename*

Description

Retreives the first 4 character strings separated by '_' from from a file path to a .RData object.

Usage

```
get_file_basename(filename)
```

Arguments

`filename`	a file name or full file path. filename must have file extension '.RData'.

Details

filename must have file extension '.RData'.

Value

Returns the first 4 character strings separated by '_' from from a file path to a .RData object.

Author(s)

Diana Hall

Examples

```
data("S") # load data
get_file_basename(S$file)
```

`get_num_ae` *get_num_ae*

Description

Adds a field to a 's' spike object 'nae' that lists for each well the # of active electrodes (electodes firing > 5spike/minute).

Usage

```
get_num_ae(s2)
```

Arguments

`s2`	an 's' object containing spike trains, channel names, etc.

Value

returns 'nae' field in 's' which is a vector of # of active electrodes (electodes firing > 5 spikes/minute). Each vector entry is named by the well to which the data corresponds.

Author(s)

Diana Hall

Examples

```
data("S") # load data
b<-get_num_ae(S)
b$nae
```

`get_project_plate_name`

get_project_plate_name

Description

returns the first portion of file .RData spike object named according to convention of Project name, experiment date (MMDDYYYY format) and plate serial number separated by a '_' as in "exampleRecording_1012016_plate1_DIV1_spike_list.csv" in data package directory.

Usage

```
get_project_plate_name(file)
```

Arguments

`file`	a full file path or file name

Value

Returns a character string of the project name, experiment date and plate serial number in a .RData file path. see example.

Examples

```
data("S") # load data
get_project_plate_name(S$file)
```

get_wt *Get WT*

Description

Extracts all treatments/genotypes and allows user to choose single treatment as wild type/reference for downstream analyses

Usage

```
get_wt(s)
```

Arguments

s	MEA dataframe structure

Value

A string corresponding to the user's choice

Examples

```
data("S")
s<-list()
s[[1]]<-S

wt <- get_wt(s)
```

has_network_spikes *A utility function to check if network spikes are detected.*

Description

For an returned object from calcualte.network.spikes, this function provides a utility checking if it contains any network spikes from any well.

Usage

```
has_network_spikes(nspikes)
```

Arguments

nspikes	The network spike object returned by calling calcualte.network.spikes.

Value

Return a boolean value indicating if any network spikes are found from the network spikes object.

isi *isi*

Description

calculates the isi (inter-spike interval) (s) between sucessive spikes in a input spike train.

Usage

```
isi(train)
```

Arguments

train	spike train: a set of non-decreasing timestamps (s)

Value

a vector of isis: first entry is ISI between first & second spike in input spike train and so forth. Total length is 1 less than input spike train.

Author(s)

Diana Hall

Examples

```
data("S") # load data
b<-isi(S$spikes[[1]])
S$spikes[[1]][1:4]
b[1:3]
```

load_spikelist *Load Robject File*

Description

Loads a previously saved Rdata file of a recording.

Usage

```
load_spikelist(spk_data_file)
```

Arguments

spk_data_file	MEA recording Rdata file

Value

loaded R object

`mi_find_bursts` *Find bursts*

Description

For one spike train, find the bursts using the maximum interval method.

Usage

```
mi_find_bursts(spikes,mi_par)
```

Arguments

`spikes`	A spike train of one channel, located in MEA data structure (example S$spikes[[1]]).
`mi_par`	A list of burst features: **beg_isi** Beginning inter spike interval **end_isi** Ending inter spike interval **min_ibi** Minimum inter burst interval to combine bursts **min_durn** Minimum duration to consider as burst **min_spikes** Minimum spikes to consider as burst

Value

Returns a matrix of burst information for a specific channel. Matrix columns are:

`beg`	the number of spike that is first in the burst
`end`	number of the last spike in the burst
`ibi`	time interval from previous burst
`durn`	durarion of burst in seconds
`mean_isis`	average inter spike interval within the burst
`si`	surprise index, allways 1 for mi algorithm

Author(s)

Stephen Eglen

References

Eytan and Marom (2006) J Neuroscience.

Examples

```
data("S")
allb <- lapply(S$spikes, mi_find_bursts, S$parameters$mi_par )
```

`nb_matrix_to_feature_dfs` *Convert network burst data matrix to a list of data frames.*

Description

Convert network burst data matrix to a list of dataframes. Each dataframe has rows representing wells while columns representing different timepoints(DIVs). The dataframe format alllows well level permutaiton based tests to be done much easier.

Usage

```
nb_matrix_to_feature_dfs(matrix_and_feature_names)
```

Arguments

`matrix_and_feature_names`
: The data matrix return by calling function calculate_network_bursts.

Value

Returns a list of dataframes, ith each representing a feature matrix, with rows for wells and columns for different timepoints(DIVs).

Author(s)

Quanli Wang

See Also

`calculate_network_bursts`

`parameters` *A list of parameters with default values that user can customize.*

Description

A list of parameters with default values that user can customize.

Usage

```
data("parameters")
```

Format

The format is: List of 20 $ spike.csv : logi TRUE $ spike.plot : logi TRUE $ burst.csv : logi TRUE $ burst.plot : logi TRUE $ burst_type : chr "mi" $ s_min : num 5 $ ns.csv : logi TRUE $ ns.plot : logi TRUE $ elec_min_rate : num 0.0167 $ elec_max_rate : num 1000 $ well_min_rate : num 0 $ mi_par :List of 5 ..$ beg_isi : num 0.1 ..$ end_isi : num 0.25 ..$ min_ibi : num 0.8 ..$ min_durn : num 0.05 ..$ min_spikes: num 5 $ ns_t : num 0.01 $ ns_n : num 3 $ sur : num 100 $ burst_distribution_ibi :List of 7 ..$ perform : num 1 ..$ min_cases : num 15 ..$ x_axis_lim : num 20 ..$ bins_in_sec : num 5 ..$ min_values : num 0 ..$ filter_by_min: num 0 ..$ per_well : num 0 $ burst_distribution_durn :List of 7 ..$ perform : num 1 ..$ min_cases : num 15 ..$ x_axis_lim : num 18 ..$ bins_in_sec : num 10 ..$ min_values : num 0 ..$ filter_by_min: num 0 ..$ per_well : num 0 $ burst_distribution_isi :List of 7 ..$ perform : num 1 ..$ min_cases : num 15 ..$ x_axis_lim : num 0.5 ..$ bins_in_sec : num 100 ..$ min_values : num 0 ..$ filter_by_min: num 0 ..$ per_well : num 0 $ burst_distribution_nspikes :List of 7 ..$ perform : num 1 ..$ min_cases : num 5 ..$ x_axis_lim : num 200 ..$ bins_in_sec : num 1 ..$ min_values : num 0 ..$ filter_by_min: num 0 ..$ per_well : num 0 $ burst_distribution_spike_freq:List of 7 ..$ perform : num 1 ..$ min_cases : num 15 ..$ x_axis_lim : num 300 ..$ bins_in_sec : num 1 ..$ min_values : num 0 ..$ filter_by_min: num 0 ..$ per_well : num 0

Examples

```
data(parameters)
```

---

permute_features_and_plot *Write PDF*

Description

Generates a PDF containing plots and p-values for each feature. P-values are generating using Mann Whitney and permutation tests. This function requires that you create a list of dataframes for a given feature type (e.g. spikes) using the aggregate.data() function

Usage

```
permute_features_and_plot(s, wt, np, features_list, type, output_dir)
```

Arguments

s	MEA data structure
wt	The treatment that will act as the wildtype/reference for the Mann Whitney and Permutation tests
np	Number of permutations to be performed
features_list	A list of dataframes containing data for a given feature
type	Type of features contained in features_list (e.g. spikes, ns, or bursts)
output_dir	Directory where output files will be generated

Value

A PDF file containing the plots and p-values.

Author(s)

Ryan Dhindsa

Examples

```
data("S")
spike_features<-aggregate_features(S, feat_type="spike" )
wt <- "untreated"
output_dir = getwd()
permute_features_and_plot(S, wt, np, spike_features, "spikes", output_dir)
```

---

plot.sttcp *Simple plotting function to show STTC profile*

Description

Simple plotting function to show STTC profile

Usage

```
## S3 method for class 'sttcp'
plot(x, ...)
```

Arguments

x	An sttcp object generated by the sttcp() function.
...	Other plotting arguments to pass to plot()

Details

This is a simple plotting function to show the STTC profile. Examples of its usage are given in the help page for sttcp.

Value

Nothing is returned.

Author(s)

Stephen Eglen

See Also

sttcp

plot_active_wells_network_spikes
plot_active_wells_network_spikes

Description

Plots related to network spike for each well with network spikes in format of users choosing.

Usage

```
plot_active_wells_network_spikes(nspikes)
```

Arguments

nspikes	list of attributes related to network spikes: wells, plate layout and network spike information for each well. See calculate_network_spikes for further details.

Value

returns a multi-page plot.

See Also

calculate_network_spikes xyplot_network_spikes

Examples

```
data("S")
data('parameters')
nspikes <- calculate_network_spikes( S, parameters$sur ,parameters$ns_n, parameters$ns_t )

  pdf(file=NSPlotPath)
  xyplot.network.spikes(nspikes)
  plot_active_wells_network_spikes(nspikes)
  dev.off()
```

plot_mean_firingrate_by_eletrode_by_div
plot.mean.firingrate.by.eletrode.by.div

Description

Displays average firing rate by well and by electrode for each DIV available.

Usage

```
plot_mean_firingrate_by_eletrode_by_div(s)
```

Arguments

s	's' object. must be a list, with each DIV a different entry.

Examples

```
data("S")
s<-list()
s[[1]]<-S

  plot.mean.firingrate.by.eletrode.by.div(s)
```

---

plot_mean_firingrate_by_well_by_div
plot.mean.firingrate.by.well.by.div

Description

Displays average firing rate by well for each DIV available. First plot well rate in average Hz/electrode and second plot is Hz/total spikes well.

Usage

```
plot_mean_firingrate_by_well_by_div(s)
```

Arguments

s	's' object. must be a list, with each DIV a different entry.

Value

Plot is output, location and path to plot may be controlled by R's plotting apparatus e.g. 'pdf()'

Author(s)

Diana Hall

Examples

```
data("S")
s<-list()
s[[1]]<-S

  plot.mean.firingrate.by.well.by.div(s)
```

plot_network_spikes *Generic method for plotting network spikes.*

Description

The generic plotting function for network spikes.

Usage

```
plot_network_spikes(ns, ...)
```

Arguments

ns	The network spike object returned after running calculate_network_spikes.
...	Additional plotting options that is general to plot functions.

Value

None. network spikes related plots will be generated in current plotting device.

plot_plate_summary_for_bursts *Plot burst features*

Description

Plots all bursting features in a _burst_plot.pdf under the output directory.

Usage

```
plot_plate_summary_for_bursts(s, outputdir,parameters)
```

Arguments

s	MEA data structure
outputdir	Output directory
parameters	meaRtools basic parameter list

Details

The plot function will plot all the features calculated for the bursts in the recording. Those include: Mean Firing Rate by Plate (Hz), Mean firing rate, Mean Duration, Number of bursts by channel and well, Mean Inter Burst Interval, Mean ISI within bursts, Mean burst per minute, Mean spikes in a burst and % spikes in a burst. The function also calls calc_burst_distributions to calculate and plot burst feature distributions.

Value

A _burst_plot.pdf is printed under the output directory

Examples

```
data("S")
plot_plate_summary_for_bursts(S,"/Analysis")
```

`plot_plate_summary_for_spikes` *plot.plate.summary.for.spikes*

Description

Diana needs to add document here.

Usage

```
plot_plate_summary_for_spikes(s, outputdir)
```

Arguments

`s`	's' .RData object. Each DIV must constitute one entry in list.
`outputdir`	directory path where plot will be saved to.

Value

Multiple page plot in pdf format containing data on which electrodes have recorded any spikes, ISI (inter-spike interval) histogram by plate and by well by electrode, log ISI histogram by plate and by electrode, average electrode firing rate by well, & binned electrode firing rate over recording duration.

Author(s)

Diana Hall

Examples

```
data("S")
s<-list()
s[[1]]<-S

  plot.plate.summary.for.spikes(s, outputdir="/Desktop")
```

read_spikelist *Axion convert spk_list to r_object*

Description

Converts the Axion spk_list file to a Rdata object and initializes it with all spike and plate info

Usage

```
read_spikelist(key, spk_list_file, chem_info ,r_object_dir)
```

Arguments

key	base name of spk_list file
spk_list_file	The full spk_list file name (including path)
chem_info	plate layout information list as loaded using function chem_info_2
r_object_dir	Directory of r_object files

Value

save_file	Full path of the saved r_object data file

See Also

chem.info.2

Examples

```
master_chem_file<-paste0( system.file(package = "meaRtools"),"/data",
   "/exampleRecording_1012016_plate1_expLog.csv" )

spike_list_file<-paste0( system.file(package = "meaRtools"),"/data",
"/exampleRecording_1012016_plate1_DIV1_spike_list.csv" )

title<-strsplit(basename(spike_list_file), ".csv")[[1]][1]
# get plate chemical info for each file in the list

plate_chem_info<-chem_info_2( file=spike_list_file, master_chem_file = master_chem_file )

r_object_file_name<-read_spikelist(key=title,
                  spk_list_file=plate_chem_info,
                  chem_info=plate_chem_info,r_object_dir="/")
```

remove_spikes *remove_spikes*

Description

removes all spikes and associated meta data from 's' spike object except those specified by 'ids'.

Usage

```
remove_spikes(s, ids)
```

Arguments

s	's' list object, needs to contain a 'spikes' field with spike train
ids	Name or index of channel(s) to be kept, all other channels removed. either name of channel, e.g. "E5_12" or an vector of idices c(1,2) corresponding to channel index. If a negative index is given, then that channel and associated data will be removed.

Value

's' object.

See Also

construct.s

Examples

```
data("S") # load data
r<-remove_spikes(S, c(-1, -2))

S$channels[1:2] # original 's' object first 2 channels
r$channels[1:2] # first 2 channels have been removed

S$NCells # original count of channels
r$NCells # count of channels after 2 channels removed

S$nspikes # original spike count of first 2 channels
r$nspikes # spike count of first 2 channels after 2 channels removed

# OR keep only first 2 channels
t<-remove_spikes(S, c(1, 2))
t$channels
```

run_TMcpp *Compute STCC direct in Cpp*

Description

run the STTC code for a spike train (Cpp version)

Usage

```
run_TMcpp(dt, start, end, spike_times_1, spike_times_2)
```

Arguments

dt	bin width for
start	start time in seconds
end	end time in seconds
spike_times_1	spike train 1
spike_times_2	spike train 2

Details

Internal computation

Value

STTC value

Author(s)

Stephen Eglen

S *example 'S' object*

Description

An example 'S' list object containing multiple fields describing 1 minute recording on a 48 well plate.

Usage

```
data("S")
```

Format

channels electrode names

spikes a list of spike trains for each channel

nspikes # spikes for each channel

NCells total # electrodes

meanfiringrate mean firing rate by channel

file full path of file

layout electrodes grid positions for all electrodes on plate

rates list with average count and firing per time_interval (s) as well as plate average

rec_time 2 element vector of first and last spike time of recording

goodwells well names for all wells meeting minimum firing criteria

treatment treatments for each well

size chemical compound size for each treatment

units units of dose of treatment

dose dose of treatment

well well names

nae # active electrodes (firing>5spikes/min)

cw wells that each channel belongs to

parameters A list of parameters, see data("parameters")

allb for each electrode, a matrix of burst related information

bs burst summary, a data frame containing burst endpoints by electrode

ns_all for each well, a list of network spike information

isis list of inter-spike interval (isi) (s) by channel

mean_isis list of average isi by channel

sd_isis list of standard deviation of isi by channel

well_stats data frame containing well level firing rate information

Details

Created by use of functions available in package.

Examples

```
data('S')
names(S)
```

si_find_bursts *Find bursts*

Description

For one spike train, find the bursts using the Poisson surprise method.

Usage

```
si_find_bursts(spikes,s_min,burst_isi_max)
```

Arguments

spikes	A spike train of one channel, located in MEA data structure (example S$spikes[[1]]).
s_min	A minimum value for the surprise index
burst_isi_max	ISI threshold used by the the surprise index algorithm

Value

Returns a matrix of burst information for a specific channel. Matrix columns are:

beg	the number of spike that is first in the burst
end	number of the last spike in the burst
ibi	time interval from previous burst
durn	durarion of burst in seconds
mean_isis	average inter spike interval within the burst
si	surprise index

Author(s)

Stephen Eglen

References

Eytan and Marom (2006) J Neuroscience.

Examples

```
data("S")
allb <- lapply(S$spikes, si_find_bursts, S$parameters$s_min )
```

`sttc` *Compute STTC for a pair of spike trains*

Description

Compute STTC for a pair of spike trains

Usage

```
sttc(a, b, dt = 0.05, rec_time = NULL)
```

Arguments

`a`	first spike train
`b`	second spike train
`dt`	bin size in seconds
`rec_time`	2-element vector: start and end time

Details

The Spike Time Tiling correlation (STTC) is computed for a pair of spike trains. The method is defined in Cutts and Eglen (2014). We assume that the spike trains are ordered, smallest-time first.

Value

STTC a scalar bounded between -1 and +1.

Author(s)

Stephen J Eglen

Examples

```
a = c(1, 2, 3, 4, 5)
b = a+0.01
c = a+0.5
sttc(a, b)==1
sttc(a, c)==0
```

`sttcp` *Compute*

Description

Compute STTC profile for a pair of spike trains

Usage

```
sttcp(a, b, dt = 0.05, tau_max = 5, tau_step = 0.1, beg = NULL,
  end = NULL)
```

Arguments

`a`	spike train 1
`b`	spike train 2
`dt`	time window for STTC
`tau_max`	maximum time shift
`tau_step`	step size in tau
`beg`	start of recording. When NULL use the minimum spike time from the two trains.
`end`	end of recording. When NULL use the maximum spike time from the two trains.

Details

We extend the STTC to a profile (or correlogram) by shifting one spike train by amount tau, where tau varies in [-tau_max, +tau_max] in steps of tau_step.

Value

List containing the STTC profile.

Author(s)

Stephen Eglen

Examples

```
t1 <- -cumsum(log(runif(1000)) / 2)
t2 <- -cumsum(log(runif(1000)) / 2)
corr <- sttcp(t1, t2)
plot(corr, main="cross correlation")
autocorr <- sttcp(t1, t1)
plot(autocorr, main="auto correlation")
```

`sttcp_ab` *Compute STTC profile for two spike trains*

Description

Compute STTC profile for two spike trains

Usage

```
sttcp_ab(a, b, start, end, dt, tau_sep, tau_max)
```

Arguments

`a`	Spike train 1
`b`	Spike train 2
`start`	Start time
`end`	End time
`dt`	coincidence window for STTC
`tau_sep`	step size for tau in [-tau_max, +tau_max]
`tau_max`	maximum tau value

Details

Compute the STTC profile for two spike trains using C++.

Value

obj An object of type "sttcp", containing the tau values and correlations.

Author(s)

Stephen Eglen

`sttc_allspikes1` *Compute STTC for all pairs of spike trains*

Description

Compute STTC for all unique pairs of spike trains

Usage

```
sttc_allspikes1(spikes, dt, beg, end)
```

Arguments

spikes	List of spike trains
dt	tiling window
beg	start time
end	end time

Details

Return a matrix of all STTC values

Value

Matrix of STTC values. Upper diagonal matrix only; diagonal elements should be 1.

Author(s)

Stephen Eglen

---

summarize_network_spikes

Generate network spikes based features.

Description

This function takes the returned object from calcualte.network.spikes function and parse and filter them using customed filters to regenerate features used by IGM MEA projects.

Usage

```
summarize_network_spikes(e, nspikes, ns_e, sur)
```

Arguments

e	A spike list object for a set of electrodes.
nspikes	The spike list object returned from calling calculate_network_spikes.
ns_e	Minumum number of spikes for each electrode within the network spike window. Most IGM MEA projects use a value of 2.
sur	This parameter is related to the number of datapoints to be used in summmarizing mean network spikes, which will be only used for network spike diagnostics. The default value of 100 will usually be sufficient.

Value

Returns a new spikes object with filtered and re-calcualted features.

Author(s)

Quanli Wang

`tiling_correlogramcpp` *Compute all STTPs for a set of spike trains*

Description

Compute all STTPs for a set of spike trains

Usage

```
tiling_correlogramcpp(spikes, n, nspikes, first_spike, start, end, dt, tau_sep,
  tau_max)
```

Arguments

`spikes`	Concatenated list of spike trains
`n`	number of spike trains
`nspikes`	Vector containing the number of spikes in each train
`first_spike`	Index to the first spike in each train.
`start`	Start time of recording in seconds
`end`	End time of recording in seconds
`dt`	Coincidence window for STTC
`tau_sep`	Step size for taus.
`tau_max`	Maximum absolute tau value.

Details

This computes all pairwise STTPs for spike trains. (This may be of use for Tom's internal code, rather than for production use.)

Value

Pairwise STTPs for all spike trains

Author(s)

Tom Edinburgh

`tiling_correlogramcpp_index` *Compute STTPs for just two spike trains, A and B*

Description

Compute STTP for just one pair of trains

Usage

```
tiling_correlogramcpp_index(spikes, n, nspikes, first_spike, start, end, dt,
  tau_sep, tau_max, a, b)
```

Arguments

`spikes`	Concatenated list of spike trains
`n`	number of spike trains
`nspikes`	Vector containing the number of spikes in each train
`first_spike`	Index to the first spike in each train.
`start`	Start time of recording in seconds
`end`	End time of recording in seconds
`dt`	Coincidence window for STTC
`tau_sep`	Step size for taus.
`tau_max`	Maximum absolute tau value.
`a`	Number of first spike train
`b`	Number of second spike train

Details

If you have the spikes from an array in a flattened form, you can compute the STTP for just two of the spike trains, rather than computing all pairwise STTPs.

Value

STTP for spike trains a and b

Author(s)

Tom Edinburgh

write_features_to_files

Write feature data to an output file

Description

Takes in list of dataframes (one per feature) from an MEA data structure that is produced by aggregate_features and writes output to Files. Each dataframe corresponds to one feature, containing values for each well across each DIV of recording

Usage

```
write_features_to_files(s, features_list, output_dir, type)
```

Arguments

s	MEA data structure
features_list	list of dataframes, one for each feature.
output_dir	Output directory)
type	Type of features (e.g. "spikes", "ns", "bursts")

Value

Write one csv per feature for the feature type requested.

Examples

```
data("S")
s<-list()
s[[1]]<-S
  spike_features = aggregate_features(s, "spike")

    write_features_to_files(s, spike_features, analysis$output_dir, "spikes")
```

write_network_spikes_to_csv

Summarize and write netowrk spikes features into a csv file.

Description

Summarize and write netowrk spikes features into a csv file.

Usage

```
write_network_spikes_to_csv(s, nspikes, outputdir)
```

Arguments

s	A list of MEA recordings, typically from the same MEA plate at different time point.
nspikes	The spike list object returned from calling calculate_network_spikes.
outputdir	The user defined output directory while the cvs file to be writen. There should not have a file sperator at the end of the outputdir.

Value

None.

---

write_plate_summary_for_bursts

Prints bursting features

Description

The function reads the MEA data structure and uses the 'allb' list built using mi_find_bursts. It then prints all bursting features summary per well and per channel in _bursts.csv and _well_bursts.csv

Usage

```
write_plate_summary_for_bursts(s, outputdir)
```

Arguments

s	MEA data structure
outputdir	Output directory

Value

Output file _bursts.csv holds all features generated for bursts per well and per channel:

treatment	the treatment/genotype based on the experimental log file plan
well	well number
nae	number of active electrodes
nAB	number of electrodes with bursts
duration	total duration of bursts
mean_dur	mean duration of bursts
mean_freq	firing rate (Hz)
nbursts	number of bursts
bursts_per_sec	bursts/second.matrix(nrow=0,ncol=1)
bursts_per_min	bursts/min
sd_dur	sd of burst duration
mean_freq_in_burst	average frequency of spikes in a burst

sd_freq_in_burst
: sd of frequency of spikes in a burst

mean_spikes_in_burst
: mean number of spikes in a burst

sd_spikes_in_burst
: sd of number of spikes in a burst

total_spikes_in_burst
: total number of spikes in a bursts

per_spikes_in_burst
: percent of spikes in a burst

mean_isis
: mean ISI within a burst

sd_isis
: sd ISI within a burst

mean_ibis
: mean IBI

sd_ibis
: sd of ibis

cv_ibis
: Coefficient of variation of IBI (= mean_ibi/sd_ibi)

file
: input recording file

Examples

```
data("S")
d<-dir.create(paste0(getwd(),"/Analysis") )
s<-list(); s[[1]]<-S
write_plate_summary_for_bursts(s, paste0(getwd() ) )
```

---

write_plate_summary_for_spikes

write_plate_summary_for_spikes

Description

Produces csv output related to firing rate by DIV to directory of user specified output directory

Usage

```
write_plate_summary_for_spikes(s, outputdir)
```

Arguments

s
: 's' spike .RData object. Must be a list with one entry per DIV.

outputdir

Value

One .csv file for each DIV is output and one additional file comprising all DIVs. Quantification of activity levels including total spike count, well and elctrode level firing rate, as well as ISI and standard deviation of applicable features.

Author(s)

Diana Hall

Examples

```
data("S")
s<-list()
s[[1]]<-S

  path<-system.file()
  write_plate_summary_for_spikes(s , path)
```

`xyplot_network_spikes` *xyplot for network spikes at the plate level.*

Description

xyplot for network spikes at the plate level. It will detech the plate layout and plot individual wells according to plate layout.

Usage

```
xyplot_network_spikes(nspikes)
```

Arguments

`nspikes`	The returned obect from calling summary.network.spikes.

Value

Return the handle of xyplot.

Author(s)

Quanli Wang

Index