Package ‘telemetR’

May 8, 2026

Type Package

Title Filter and Analyze Generalised Telemetry Data from Organisms

Version 1.0

Description Analyze telemetry datasets generalized to allow any technology. The filtering steps check for false positives caused by reflected transmissions from surfaces and false pings from other noise generating equipment. The filters are based on JSATS filtering algorithms found in package 'filteRjsats' <https://CRAN.R-project.org/package=filteRjsats> but have been generalized to allow the user to define many of the filtering variables. Additionally, this package contains scripts used to help identify an optimal maximum blanking period as defined in Capello et al (2015) <doi:10.1371/journal.pone.0134002>. The functions were written according to their manuscript description, but have not been reviewed by the authors for accuracy. It is included here as is, without warranty.

License GPL (>= 3)

Encoding UTF-8

LazyData true

LazyDataCompression xz

Imports dplyr, tidyr, lubridate, zoo, ggplot2

Suggests knitr, rerddap, rmarkdown

RoxygenNote 7.2.3

Depends R (>= 4.1)

VignetteBuilder knitr

NeedsCompilation no

Author Taylor Spaulding [aut, cre] (ORCID: <https://orcid.org/0000-0003-4381-5296>)

Maintainer Taylor Spaulding <tspaulding@esassoc.com>

Repository CRAN

Date/Publication 2023-04-13 11:40:02 UTC

Contents

add_org . . . 2
blanked_detects . . . 3
blanking_event . . . 4
build_residence . . . 6
conv_thresh . . . 7
conv_thresholds . . . 7
dat_filt1 . . . 8
dat_orgfilt . . . 9
duration_compare . . . 10
ex_opt . . . 11
ex_rSSR . . . 11
filtered_detections . . . 12
filter_2h . . . 13
filter_4h . . . 14
fish . . . 14
format_detects . . . 15
format_org . . . 16
format_receivers . . . 18
opt_mbp . . . 19
prefilter . . . 20
raw_detections . . . 21
receivers . . . 22
reftags . . . 23
renorm_SSR . . . 23
residence_plot . . . 24
rSSR_plot . . . 24
setup_blanking . . . 25
Set_GVs . . . 26
time_test . . . 27

Index **28**

`add_org` *Add Organism Data to a Detection Dataframe*

Description

This function takes a prefiltered detection dataframe from ‘prefilter()‘ and joins it to organism data formatted using the ‘format_org()‘ function. Detections are then filtered further based on the date and time of tag release and expected battery life. Detections occurring before release of the tag or after 2x the expected battery life are removed.

Usage

```
add_org(prefilter_file, org, time_before_detection, time_unit)
```

Arguments

`prefilter_file` a prefiltered detection dataframe from ‘prefilter()‘

`org` a dataframe of organism data retrieved from ‘get_org_data()‘ or ‘format_org()‘

`time_before_detection` How long before detection could an organism be released and still detected? Generally 2x the expected tag life.

`time_unit` The unit of time used for time_before_detection (seconds, minutes, hours, days, weeks, months)

Value

A filtered dataframe converting the raw detection data into rows of detections

Examples

```
# Format the organism data
formatted_fish <- format_org(data = fish,
                             var_Id = "TagCode",
                             var_release = "Release_Date",
                             var_tag_life = "TagLife",
                             var_ping_rate = "PRI",
                             local_time_zone = "America/Los_Angeles",
                             time_format = "%Y-%m-%d %H:%M:%S")

# Add organism data to the prefiltered detection data
add_org(prefilter_file = dat_filt1,
        org = formatted_fish,
        time_before_detection = 120,
        time_unit = "days")
```

---

blanked_detects *Example Multi-blanked Detection Data*

Description

An example dataset of real acoustic telemetry detections of fish at several receivers within the California Central Valley from 2021. These detections have already been been processed using ‘blanking_event()‘ to create events using maximum blanking periods from 3 to 1,500 seconds to reprocess the data. Each row represents a single event which includes >=1 detection(s) per fish per site which occur within the specified maximum blanking period ‘mbp_n‘.

Usage

```
blanked_detects
```

Format

‘blanked_detects‘ A data frame with 44,630 rows and 9 columns:

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

Tag_Code The hexadecimal acoustic tag ID code

mbp_n the maximum blanking period used to create this event

event_change An increasing number which identifies the event number; one event per fish per site for all detections which occur within ‘mbp_n‘ seconds of the next.

receiver_general_location The more general geographic name of the location of the receiver

start_time The Date and Time of the first detection within the event

end_time The Date and Time of the last detection within the event

n_det The total number of detections contained within the event

duration the total length of time of the event in seconds

`blanking_event` *Create Potential Blanking Periods for Identifying Optimal Blanking Period*

Description

Takes a dataframe with telemetry detection data and a list of potential Blanking Period multipliers (n_val) and crosses them, duplicating the entire dataframe by the length of n_val. Detections are grouped by individual, site, and any supplied grouping variables. Then events are created by collecting detections which occur within n_val*ping_rate from the next detection. This function can be very slow depending on the size of the dataframe.

Usage

```
blanking_event(
  data,
  var_site,
  var_Id,
  var_datetime,
  var_groups = NULL,
  var_ping_rate,
  n_val,
  time_unit
)
```

Arguments

`data`	the detection dataframe with columns for sites, tag IDs, datetime, any grouping variables, and the expected ping rate.
`var_site`	the column name, in quotes, which identifies unique residency sites, these sites should be as distinct as possible, such that it is infrequent that organisms can be detected at two sites at a given time.
`var_Id`	the column name, in quotes, which identifies the individual transmitter/tag/organism identifier.
`var_datetime`	the column name, in quotes, which identifies the date and time of the detection event. This column should already have been converted to POSIXct format.
`var_groups`	a single string or vector of strings of the columns which should be used to group animals. Common groupings are species and cohorts.
`var_ping_rate`	the column name, in quotes, which identifies the temporal frequency at which the transmitter emits a detectable signal.
`n_val`	a vector sequence of integers which can be multiplied by the ping rate to construct multiple potential blanking periods. The range and step values for n should be selected based on prior knowledge about general behavior habits of the study organism and the functionality of the equipment. For more information, please refer to Capello et. al. 2015.
`time_unit`	the preferred unit of time to calculate durations, this should correspond to the ping_rate, (i.e. if the ping rate is 3 seconds, the preferred time_unit is seconds). If the preferred time_unit is on the same scale as the ping_rate, the ping rate should be converted to the same scale.

Value

A dataframe which has been crossed with all integers in n_val, and which has been condensed into events. Please refer to Capello et. al. 2015 for further detail about the creation of these events.

Examples

```
# Create a dataframe of events blanked by a set of n_values from 1:10
blanking_event(data = filtered_detections,
               var_Id = "Tag_Code",
               var_site = "receiver_general_location",
               var_datetime = "DateTime_Local",
               var_groups = "fish_type",
               var_ping_rate = "tag_pulse_rate_interval_nominal",
               n_val = c(1:2),
               time_unit = "secs")
```

build_residence — *Build Continuous Residence Events*

Description

Takes a dataframe with telemetry detection data and a single optimum blanking period chosen from the output of opt_mbp(), and groups detections by individual, site, and any supplied grouping variables into residence events. The residence events are created by collecting detections which occur within the selected optimum maximum blanking period from the next detection. This function can be very slow depending on the size of the dataframe.

Usage

```
build_residence(
  data,
  var_groups,
  var_Id,
  var_datetime,
  var_site,
  opt_mbp,
  time_unit
)
```

Arguments

`data`	the detection dataframe with columns for sites, tag IDs, datetime, any grouping variables, and the expected ping rate.
`var_groups`	a single string or vector of strings of the columns which should be used to group animals. Common groupings are species and cohorts.
`var_Id`	the column name, in quotes, which identifies the individual transmitter/tag/organism identifier.
`var_datetime`	the column name, in quotes, which identifies the date and time of the detection event. This column should already have been converted to POSIXct format.
`var_site`	the column name, in quotes, which identifies unique residency sites, these sites should be as distinct as possible, such that it is infrequent that organisms can be detected at two sites at a given time.
`opt_mbp`	a single optimum blanking period chosen from the output of opt_mbp()
`time_unit`	the unit of time used by the optimum maximum blanking period, often on the same scale as the ping rate for the transmitter.

Value

A dataframe of detections which has been condensed into continuous residence events based on the optimum maximum blanking period selected.

Examples

```
# Build a set of detection events after determining the optimal blanking
# period (e.g. 2500 seconds)
build_residence(data = filtered_detections,
                var_groups = "fish_type",
                var_Id = "Tag_Code",
                var_datetime = "DateTime_Local",
                var_site = "receiver_general_location",
                opt_mbp = 2500,
                time_unit = "secs")
```

---

conv_thresh *Example 95 Percent Convergence Threshold*

Description

Example output from the ‘conv_thresholds()‘ function, calculating the 95 convergence thresholds for the rSSR data found in ‘ex_rSSR‘.

Usage

```
conv_thresh
```

Format

‘conv_thresh‘ A data frame with 1 rows and 5 columns:

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

min_val The minimum rSSR value

max_val The maximum rSSR value

threshold the rSSR value which represents the ‘thresh_level‘ cutoff for estimating convergence

thresh_level The desired convergence level (100-x)

---

conv_thresholds *Calculate Convergence Thresholds for the rSSR curve*

Description

Takes a dataframe created by renorm_SSR and calculates the range in values and then calculates thresholds given. Suggested values are 0.5, 0.1, and 0.005. The rSSR calculated for each MBP should decrease with each increasing blanking period until they reach close to zero, which We consider convergence. Since the rSSR curve generally bounces around an assymptote and often does not reach or stay at 0, we set a threshold a priori for identifying convergence.

Usage

```
conv_thresholds(rSSR_df, var_groups, thresh_levels = c(0.05, 0.01, 0.005))
```

Arguments

rSSR_df	a dataframe created by created by renorm_SSRduration compare showing the renormalized sum of squares of the residuals between one potential blanking period and the next.
var_groups	a single string or vector of strings of the columns which should be used to group organisms. Common groupings are species and cohorts.
thresh_levels	a single value or vector of values used to set thresholds for identifying convergence.

Value

A dataframe of rSSR values corresponding to the given convergence threshold

Examples

```
# Calculate the 95% "convergence" threshold for the rSSR data
conv_thresholds(rSSR_df = ex_rSSR,
                var_groups = "fish_type",
                thresh_levels = 0.05)
```

---

dat_filt1 *Prefiltered detection data*

Description

An example dataset of real acoustic telemetry detections of fish at several receivers within the California Central Valley from 2021. These detections have already been been processed using ‘prefilter()‘ from this package or companion package ‘filteRjsats‘.

Usage

```
dat_filt1
```

Format

‘dat_filt1‘ A data frame with 47,931 rows and 4 columns:

ReceiverSN The serial number of the detecting receiver

DateTime_Local the local time of the detection (tz = America/Los_Angeles)

Tag_Code The hexadecimal acoustic tag ID code

CheckMBP A calculated field from the prefilter checking the time between acoustic transmissions from the same tag was >0.3secs

Source

Data collected by the California Department of Water Resources 2021

`dat_orgfilt` *Filtered detection data with Organism Data*

Description

An example dataset of real acoustic telemetry detections of fish at several receivers within the California Central Valley from 2021. These detections have already been been processed using ‘prefilter()‘ and ‘add_org()‘.

Usage

```
dat_orgfilt
```

Format

‘dat_orgfilt‘ A data frame with 47,343 rows and 16 columns:

ReceiverSN The serial number of the detecting receiver

DateTime_Local the local time of the detection (tz = America/Los_Angeles)

Tag_Code The hexadecimal acoustic tag ID code

CheckMBP A calculated field from the prefilter checking the time between acoustic transmissions from the same tag was >0.3secs

TagInFile A calculated field from the add_fish filter which queries whether the tag code of the detection is associated with an organism.

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

org_release_Date The release date and time of the fish

release_location The coded name of the release site

length The length of the fish in millimeters

weight The weight of the fish in grams

tag_weight The weight of the implanted acoustic tag

tag_model The model number of the implanted acoustic tag

tag_pulse_rate_interval_nominal The pulse rate interval (time between transmissions) of the implanted tag, as reported by the manufacturer

tag_life The expected number of days the tag should continue to transmit, as reported by the manufacturer

CheckDT A calculated field which checks whether the detection occurred after the release of the fish

CheckBattLife A calculated field which checks whether the detection occurred before the tag battery is expected to expire (2x tag life)

Source

Data collected by the California Department of Water Resources 2021

`duration_compare` *Compare the duration of Potential Blanking Periods*

Description

Takes a dataframe of detection data which has been condensed by potential blanking periods generated by ‘blanking_event()‘ and compares the duration of each event to a common sequence of increasing times. If the event is longer than the duration it is flagged as "survived". The proportion of events which "survive" for each potential blanking period at each time (t) is then calculated.

Usage

```
duration_compare(event_dur, var_groups = NULL, time_seq)
```

Arguments

`event_dur`	the detection dataframe which has been condensed into discrete events using each potential blanking period.
`var_groups`	a single string or vector of strings of the columns which should be used to group organisms. Common groupings are species and cohorts.
`time_seq`	a vector of times on the same scale as the ping rate. The largest value of the sequence should be greater that the longest duration produced using blanking event, and the smallest should be shorter than the smallest blanking period.

Value

A dataframe which contains the proportion of "survived" events created by each potential blanking period for each time (t).

Examples

```
# Compare the durations of blanked detection events
duration_compare(event_dur = blanked_detects,
                 var_groups = "fish_type",
                 time_seq = c(1:10))
```

ex_opt — *Example Optimum Maximum Blanking Period*

Description

Example output from the ‘opt_mbp()‘ function, finding the optimal mbp for each group and desired convergence threshold.

Usage

```
ex_opt
```

Format

‘ex_opt‘ A data frame with 1 rows and 5 columns:

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

min_val The minimum rSSR value

max_val The maximum rSSR value

threshold the rSSR value which represents the ‘thresh_level‘ cutoff for estimating convergence

thresh_level The desired convergence level (100-x)

opt_mbp The identified optimum mbp for the given threshold and group

ex_rSSR — *Example Renormalized Sum of Squares*

Description

Example output from the ‘renorm_SSR()‘ function, calculating the renormalized sum of squares for the "survival" data found in ‘time_test‘.

Usage

```
ex_rSSR
```

Format

‘ex_rSSR‘ A data frame with 100 rows and 5 columns:

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

mbp_n The maximum blanking period (in seconds) used to create a set of events

SSR The sum of squared residuals between this ‘mbp_n‘ and the next

n the total number of events created with this ‘mbp_n‘

rSSR the renormalized sum of squared residuals between this ‘mbp_n‘ and the next

filtered_detections *Example Completely Filtered Detection Data*

Description

An example dataset of real acoustic telemetry detections of fish at several receivers within the California Central Valley from 2021. These detections have already been been processed using ‘prefilter()‘ and ‘add_org()‘.

Usage

```
filtered_detections
```

Format

‘filtered_detections‘ A data frame with 41,000 rows and 26 columns:

ReceiverSN The serial number of the detecting receiver

DateTime_Local the local time of the detection (tz = America/Los_Angeles)

Tag_Code The hexadecimal acoustic tag ID code

CheckMBP A calculated field from the prefilter checking the time between acoustic transmissions from the same tag was >0.3secs

TagInFile A calculated field from the add_fish filter which queries whether the tag code of the detection is associated with an organism.

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

org_release_Date The release date and time of the fish

release_location The coded name of the release site

length The length of the fish in millimeters

weight The weight of the fish in grams

tag_weight The weight of the implanted acoustic tag

tag_model The model number of the implanted acoustic tag

tag_pulse_rate_interval_nominal The pulse rate interval (time between transmissions) of the implanted tag, as reported by the manufacturer

tag_life The expected number of days the tag should continue to transmit, as reported by the manufacturer

CheckDT A calculated field which checks whether the detection occurred after the release of the fish

CheckBattLife A calculated field which checks whether the detection occurred before the tag battery is expected to expire (2x tag life)

dep_id A unique id is created for each receiver deployment

Make The brand of the acoustic receiver

latitude The decimal degree latitude (WGS1984) of the acoustic receiver at deployment

longitude The decimal degree longitude (WGS1984) of the acoustic receiver at deployment

receiver_location The site name of an individual receiver, often more than one ‘receiver_location‘ is found at a ‘receiver_general_location‘

receiver_general_location The more general geographic name of the location of the receiver

receiver_river_km The number of river kilometers the receiver is from the Golden Gate Bridge

receiver_start The start time of the reciever (generally when it was deployed)

receiver_end The end time of the receiver (generally when it was retrieved)

Source

Data collected by the California Department of Water Resources 2021

`filter_2h` *Basic Two Hit Filter for Detections*

Description

This function takes a detection dataframe generated from the add_org() function and filters it a second time to remove any remaining multipath detections, and then check the remaining detections by comparing the time between each detection to ensure it is less 4x the stated pulse rate interval. Called by second_filter_2h4h().

Usage

```
filter_2h(org_file, time_unit, multipath_time, org_ping_rate)
```

Arguments

`org_file`	a dataframe of detections retrieved from add_org()
`time_unit`	The unit of time used for analyses (seconds, minutes, hours, days, weeks, months)
`multipath_time`	A numeric maximum amount of time which must pass between detections for a detection to be considered a "true", not a bounced, signal.
`org_ping_rate`	The expected time between transmissions emitted from tags/transmitters implanted or attached to an organism

Value

A dataframe which has been filtered to remove false positives

Examples

```
# Apply a 2-hit filter to data previously prefiltered and with organism data
filter_2h(org_file = dat_orgfilt,
          time_unit = "secs",
          multipath_time = 0.3,
          org_ping_rate = 3)
```

filter_4h *Basic Four Hit Filter for Detections*

Description

This function takes a detection dataframe generated from the ‘add_org()‘ function and filters it a second time to remove any remaining multipath detections, and then check the remaining detections by comparing the time between detections, for a rolling window of 4 detections to ensure it is less 16.6x the stated pulse rate interval. Called by ‘second_filter()‘.

Usage

```
filter_4h(org_file, time_unit, multipath_time, org_ping_rate)
```

Arguments

org_file	a dataframe of detections retrieved from ‘add_org()‘
time_unit	The unit of time used for analyses (secs, mins, hours, days, weeks)
multipath_time	A numeric maximum amount of time which must pass between detections for a detection to be considered a "true", not a bounced, signal.
org_ping_rate	The expected time between transmissions emitted from tags/transmitters implanted or attached to an organism

Value

A dataframe which has been filtered to remove false positives

Examples

```
# Apply a 4hit filter to data previously prefiltered and with organism data
filter_4h(org_file = dat_orgfilt,
          time_unit = "secs",
          multipath_time = 0.3,
          org_ping_rate = 3)
```

fish *Fish Data*

Description

An example dataset of real fish tagged with acoustic telemetry tags and released within the California Central Valley in 2021 and 2022.

Usage

```
fish
```

Format

‘fish‘ A data frame with 7,240 rows and 60 columns:

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

TagCode The hexadecimal code of the implanted acoustic tag

Release_Date The release date and time of the fish

release_location The coded name of the release site

length The length of the fish in millimeters

weight The weight of the fish in grams

tag_weight The weight of the implanted acoustic tag

tag_model The model number of the implanted acoustic tag

PRI The pulse rate interval (time between transmissions) of the implanted tag, as reported by the manufacturer

TagLife The expected number of days the tag should continue to transmit, as reported by the manufacturer

Source

<https://oceanview.pfeg.noaa.gov/CalFishTrack/pageRealtime_download.html>

`format_detects` *Format Detections for filteRjsats*

Description

This function takes a detection dataframe from a single receiver and reformats specific columns so that they can be read by the filtering functions in filteRjsats package

Usage

```
format_detects(
  data,
  var_Id,
  var_datetime_local,
  var_frequency = NULL,
  var_receiver_serial,
  var_receiver_make = NULL,
  local_time_zone,
  time_format
)
```

Arguments

- `data` — the detection dataframe with columns for individual receivers, tag IDs,datetime, and the expected ping rate.
- `var_Id` — the column name, in quotes, which identifies the individual transmitter/tag/organism identifier.
- `var_datetime_local` — the column name, in quotes, which identifies the date and time of the detection event. This column should already have been converted to POSIXct format and should be converted to the local timezone.
- `var_frequency` — the column name, in quotes, which identifies the maximum temporal frequency at which transmitters in organisms emit a detectable signal, only for use before JSATS filtering.
- `var_receiver_serial` — the column name, in quotes, which identifies the serial number of the detection receiver
- `var_receiver_make` — the column name, in quotes, which identifies the make or brand of the detection receiver. Must be one of "ATS", "Lotek", or "Tekno", only for use before JSATS filtering.
- `local_time_zone` — the local timezone used for analyses. Uses tz database names (e.g. "America/Los_Angeles" for Pacific Time)
- `time_format` — a string value indicating the datetime format of all time fields

Value

A standardized detection dataframe which can be read by filteRjsats

Examples

```
#format the detection data
format_detects(data = raw_detections,
               var_Id = "tag_id",
               var_datetime_local = "local_time",
               var_receiver_serial = "serial",
               local_time_zone = "America/Los_Angeles",
               time_format = "%Y-%m-%d %H:%M:%S")
```

`format_org` *Format Organism Data for add_org()*

Description

This function takes a dataframe of org and tag data and renames the columns to those expected by the add_org() function

Usage

```
format_org(
  data,
  var_Id,
  var_release,
  var_tag_life,
  var_ping_rate,
  local_time_zone,
  time_format
)
```

Arguments

data	a dataframe of org and tag data
var_Id	the column name, in quotes, which identifies the individual transmitter/tag/organism identifier.
var_release	the column name, in quotes, which identifies the release date and time in POSIX format and appropriate timezone
var_tag_life	the column name, in quotes, which identified the expected tag life in days
var_ping_rate	the column name, in quotes which identifies the expected ping rate of the tag/transmitter
local_time_zone	the local timezone used for analyses. Uses tz database names (e.g. "America/Los_Angeles" for Pacific Time)
time_format	a string value indicating the datetime format of all time fields

Value

A dataframe which contains fields renamed to match those required by add_org() function

Examples

```
# Rename columns to work with functions
format_org(data = fish,
            var_Id = "TagCode",
            var_release = "Release_Date",
            var_tag_life = "TagLife",
            var_ping_rate = "PRI",
            local_time_zone = "America/Los_Angeles",
            time_format = "%Y-%m-%d %H:%M:%S")
```

format_receivers — *Format for Receiver data for filteRjsats*

Description

This function takes a dataframe of receiver metadata and reformats specific columns so that they can be read by the filtering functions in filteRjsats package

Usage

```
format_receivers(
  data,
  var_receiver_serial,
  var_receiver_make,
  var_receiver_deploy,
  var_receiver_retrieve,
  local_time_zone,
  time_format
)
```

Arguments

`data` the detection dataframe with columns for individual receivers, tag IDs,datetime, and the expected ping rate.

`var_receiver_serial` the column name, in quotes, which identifies the serial number of the detection receiver

`var_receiver_make` the column name, in quotes, which identifies the make or brand of the detection receiver. Must be one of "ATS", "Lotek", or "Tekno"

`var_receiver_deploy` the column name, in quotes, which identifies the date and time the receiver was deployed

`var_receiver_retrieve` the column name, in quotes, which identifies the date and time the receiver was retrieved

`local_time_zone` the local timezone used for analyses. Uses tz database names (e.g. "America/Los_Angeles" for Pacific Time)

`time_format` a string value indicating the datetime format of all time fields

Value

A dataframe which contains fields renamed to match those required by add_receivers() function

Examples

```
# Rename columns to work with functions
format_receivers(data = receivers,
                 var_receiver_serial = "receiver_serial_number",
                 var_receiver_make = "receiver_make",
                 var_receiver_deploy = "receiver_start",
                 var_receiver_retrieve = "receiver_end",
                 local_time_zone = "America/Los_Angeles",
                 time_format = "%m-%d-%Y %H:%M:%S")
```

`opt_mbp` *Idenitfy the Optimum MBP based on Convergence Threshold*

Description

Takes dataframes created by ‘renorm_SSR()‘ and ‘conv_thresholds()‘ and determines the corresponding "optimum" maximum blanking period (MBP) for each convergence threshold.

Usage

```
opt_mbp(rSSR_df, thresh_values)
```

Arguments

`rSSR_df`	a dataframe created by created by renorm_SSRduration compare showing the renormalized sum of squares of the residuals between one potential blanking period and the next.
`thresh_values`	a dataframe created by conv_thresholds corresponding to the chosen convergence thresholds.

Value

A dataframe showing the convergence value and corresponding optimal maximum blanking period for each grouping.

Examples

```
# Determine the optimum mbp
opt_mbp(rSSR_df = ex_rSSR,
        thresh_values = conv_thresh)
```

`prefilter` *Apply the "prefilter" to a Detection Dataframe*

Description

This function takes a detection dataframe output from format_detects and filters out multipath signals (signals which are bounced off of surfaces, usually seen in underwater systems with hard surfaces which reflect sound) and spurious signals which do not occur within a user defined time frame of the last detection (12x the ping rate for organisms or 3x the ping rate for beacons). Following this, the dataframe is standardized so that all detection dataframes from any technology type are identical and superfluous fields are removed.

Usage

```
prefilter(
  data,
  reference_tags,
  time_unit,
  multipath_time,
  org_ping_rate,
  beacon_ping
)
```

Arguments

`data`	A dataframe which is the output from read_jstats() or format_detects()
`reference_tags`	A vector of potential reference (beacon) tag IDs
`time_unit`	The unit of time used for analyses (seconds, minutes, hours, days, weeks, months)
`multipath_time`	A numeric maximum amount of time which must pass between detections for a detection to be considered a "true", not a bounced, signal.
`org_ping_rate`	The expected time between transmissions emitted from tags/transmitters implanted or attached to an organism
`beacon_ping`	The expected time between transmissions emitted from tags/transmitters used as beacon or reference tags to check receiver functionality.

Value

A standardized detection dataframe with multipath detects removed

Examples

```
# Run the prefilter on a set of raw detection data

#format the detection data
detects_formatted <- format_detects(data = raw_detections,
                                    var_Id = "tag_id",
```

```
                                        var_datetime_local = "local_time",
                                        var_receiver_serial = "serial",
                                        local_time_zone = "America/Los_Angeles",
                                        time_format = "%Y-%m-%d %H:%M:%S")

#apply the prefilter
prefilter(data = detects_formatted,
          reference_tags = reftags,
          time_unit = "secs",
          multipath_time = 0.3,
          org_ping_rate = 3,
          beacon_ping = 30)
```

`raw_detections` *Unfiltered detection data*

Description

An example dataset of real acoustic telemetry detections of fish at several receivers within the California Central Valley from 2021. These detections have not been processed to remove false positives.

Usage

```
raw_detections
```

Format

‘raw_detections‘ A data frame with 55,736 rows and 3 columns:

serial The serial number of the detecting receiver

local_time the local time of the detection (tz = America/Los_Angeles)

tag_id The hexadecimal acoustic tag ID code

Source

Data collected by the California Department of Water Resources 2021

`receivers` *Receiver Data*

Description

An example dataset of real acoustic telemetry receivers within the California Central Valley in 2021. These receivers are only those which match the serial numbers in companion dataset ‘filtered_detections‘. This data is formatted to match the California Fish Tracking receiver metadata found here: https://oceanview.pfeg.noaa.gov/CalFishTrack/.

Usage

```
receivers
```

Format

‘receivers‘ A data frame with 7,240 rows and 60 columns:

dep_id A unique id is created for each receiver deployment

receiver_make The brand of the acoustic receiver

receiver_serial_number The serial number of the acoustic receiver

latitude The decimal degree latitude (WGS1984) of the acoustic receiver at deployment

longitude The decimal degree longitude (WGS1984) of the acoustic receiver at deployment

receiver_location The site name of an individual receiver, often more than one ‘receiver_location‘ is found at a ‘receiver_general_location‘

receiver_general_location The more general geographic name of the location of the receiver

receiver_river_km The number of river kilometers the receiver is from the Golden Gate Bridge

receiver_start The start time of the reciever (generally when it was deployed)

receiver_end The end time of the receiver (generally when it was retrieved)

Source

<https://oceanview.pfeg.noaa.gov/CalFishTrack/pageRealtime_download.html>

reftags — *Example reference tags*

Description

A vector of example reference tag codes

Usage

```
reftags
```

Format

A vector of example reference tag codes

renorm_SSR — *Calculate the Renormalized Sum of Squared Residuals*

Description

Takes a dataframe of the proportion of events created by each potential blanking period which "survived" a certain time (t) created by ‘duration_compare()‘ and calculates the sum of squares of the residuals between one potential blanking period and the next. This result is then renormalized by dividing the result by the number of events created.

Usage

```
renorm_SSR(time_df, var_groups = NULL)
```

Arguments

`time_df`	a dataframe created by duration compare showing the proportion of events created by each potential blanking period which "survived" a certain time (t)
`var_groups`	a single string or vector of strings of the columns which should be used to group organisms. Common groupings are species and cohorts.

Value

A dataframe of the renormalized sum of squared residuals between each potential blanking period and the subsequent one.

residence_plot *Residency Survival Plot*

Description

Takes a dataframe of the proportion of events created by each potential blanking period which "survived" a certain time (t) and creates a plot. Used to visually look for convergences between survival lines.

Usage

```
residence_plot(time_df, var_groups = NULL, time_unit)
```

Arguments

time_df	a dataframe created by duration compare showing the proportion of events created by each potential blanking period which "survived" a certain time (t)
var_groups	a single string or vector of strings of the columns which should be used to group organisms. Common groupings are species and cohorts.
time_unit	the unit of time used to calculate durations

Value

A plot of the proportion of events created by each potential blanking period at each time (t).

Examples

```
#Plot a comparison of the number of events longer than a given time `t`
residence_plot(time_df = time_test,
               var_groups = "fish_type",
               time_unit = "secs")
# Note: that the large number of lines extending past the largest Time
# indicates that a larger t is needed to ensure convergence
```

rSSR_plot *Plot the rSSR Curve and Convergence Thresholds and Optimum MBP*

Description

Using the dataframes produced by renorm_SSR and opt_mbp, plots the rSSR curve, and all the convergence thresholds (horizontal lines) and corresponding optimum mbps (vertical lines).

Usage

```
rSSR_plot(rSSR_df, opt_mbp_df, var_groups = NULL)
```

Arguments

rSSR_df	a dataframe created by created by renorm_SSRduration compare showing the renormalized sum of squares of the residuals between one potential blanking period and the next.
opt_mbp_df	a dataframe created by opt_mbp showing the values for the convergence thresholds and optimum mbps
var_groups	a single string or vector of strings of the columns which should be used to group organisms. Common groupings are species and cohorts.

Value

A plot of the rSSR curve, convergence thresholds, and optimum mbps

Examples

```
#plot the rSSR and log(rSSR) curves
rSSR_plot(rSSR_df = ex_rSSR,
          opt_mbp_df = ex_opt,
          var_groups = "fish_type")
```

---

setup_blanking — *Setup a Detection Dataframe for Identifying the Optimal Blanking Period*

Description

Takes a dataframe with telemetry detection data and a list of potential Blanking Period multipliers (n_val) and crosses them, duplicating the entire dataframe by the length of n_val. This function is contained in blanking event.This function can be slow depending on the size of the dataframe.

Usage

```
setup_blanking(
  data,
  var_site,
  var_Id,
  var_datetime,
  var_groups = NULL,
  var_ping_rate,
  n_val
)
```

Arguments

`data`	the detection dataframe with columns for sites, tag IDs, datetime, any grouping variables, and the expected ping rate.
`var_site`	the column name, in quotes, which identifies unique residency sites, these sites should be as distinct as possible, such that it is infrequent that organisms can be detected at two sites at a given time.
`var_Id`	the column name, in quotes, which identifies the individual transmitter/tag/organism identifier.
`var_datetime`	the column name, in quotes, which identifies the date and time of the detection event. This column should already have been converted to POSIXct format.
`var_groups`	a single string or vector of strings of the columns which should be used to group animals. Common groupings are species and cohorts.
`var_ping_rate`	the column name, in quotes, which identifies the temporal frequency at which the transmitter emits a detectable signal.
`n_val`	a vector sequence of integers which can be multiplied by the ping rate to construct multiple potential blanking periods. The range and step values for n should be selected based on prior knowledge about general behavior habits of the study organism and the functionality of the equipment. For more information, please refer to Capello et. al. 2015.

Value

A dataframe which has been crossed with all integers in n_val

Examples

```
# reduce dataframe for optimal blanking period analysis
setup_blanking(data = filtered_detections,
               var_Id = "Tag_Code",
               var_site = "receiver_general_location",
               var_datetime = "DateTime_Local",
               var_groups = "fish_type",
               var_ping_rate = "tag_pulse_rate_interval_nominal",
               n_val = c(1:3))
```

`Set_GVs` *Add in Global Variables*

Description

Sets all global variables to remove warnings in package build

`time_test` *Example Time Tests for Multi-blanked Detection Data*

Description

Example output from the ‘duration_compare()‘ function, testing the duration of detection events found in ‘blanked_detects‘.

Usage

```
time_test
```

Format

‘time_test‘ A data frame with 333,400 rows and 4 columns:

t The time (in seconds) against which the duration was compared

fish_type Generally a strain, run, and species of fish (e.g. Nimbus Fall Chinook = Fall-run Chinook Salmon from Nimbus Hatchery)

mbp_n The maximum blanking period (in seconds) used to create a set of events

prop_res The proportion of all events created with ‘mbp_n‘ which have a duration longer than time ‘t‘.

Index

∗ **datasets**
- blanked_detects, 3
- conv_thresh, 7
- dat_filt1, 8
- dat_orgfilt, 9
- ex_opt, 11
- ex_rSSR, 11
- filtered_detections, 12
- fish, 14
- raw_detections, 21
- receivers, 22
- reftags, 23
- time_test, 27

add_org, 2

blanked_detects, 3
blanking_event, 4
build_residence, 6

conv_thresh, 7
conv_thresholds, 7

dat_filt1, 8
dat_orgfilt, 9
duration_compare, 10

ex_opt, 11
ex_rSSR, 11

filter_2h, 13
filter_4h, 14
filtered_detections, 12
fish, 14
format_detects, 15
format_org, 16
format_receivers, 18

opt_mbp, 19

prefilter, 20

raw_detections, 21
receivers, 22
reftags, 23
renorm_SSR, 23
residence_plot, 24
rSSR_plot, 24

Set_GVs, 26
setup_blanking, 25

time_test, 27