Package: bayesmove (via r-universe)

August 9, 2024

Type Package

Title Non-Parametric Bayesian Analyses of Animal Movement

Version 0.2.3

Description Methods for assessing animal movement from telemetry and biologging data using non-parametric Bayesian methods. This includes features for pre- processing and analysis of data, as well as the visualization of results from the models. This framework does not rely on standard parametric density functions, which provides flexibility during model fitting. Further details regarding part of this framework can be found in Cullen et al. (2022) <doi:10.1111/2041-210X.13745>.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0)

Imports dplyr (>= 0.8.3), furrr (>= 0.2.0), ggplot2 (>= 3.3.0), lubridate (>= 1.7.4), MCMCpack (>= 1.4.5), progress (>= 1.2.2), purrr (>= 0.3.3), Rcpp, rlang, tictoc (>= 1.0), tidyr (>= 1.0.0), magrittr, future (>= 1.15.1), progressr, shiny, dygraphs (>= 1.1.0), leaflet (>= 2.0.0), sf (>= 0.9-6)

URL https://github.com/joshcullen/bayesmove, https://joshcullen.github.io/bayesmove/

BugReports https://github.com/joshcullen/bayesmove/issues

Suggests covr, testthat, spelling, knitr, rmarkdown, ggforce, RcppArmadillo, xts, htmltools, shinythemes, DT, datamods, viridis

Language en-US

RoxygenNote 7.2.3

LinkingTo Rcpp, RcppArmadillo

Repository https://joshcullen.r-universe.dev

RemoteUrl https://github.com/joshcullen/bayesmove

RemoteRef HEAD

RemoteSha 07d20c7a7a3f2ffb2f05a430d68cff54ac2f379a

Contents

`assign_behavior` *Assign behavior estimates to observations*

Description

Assign behavior estimates to observations

Usage

```
assign_behavior(dat.orig, dat.seg.list, theta.estim.long, behav.names = NULL)
```

Arguments

`dat.orig`	A data frame that contains all of the original data for all animal IDs. Must be same as was used to originally segment the tracks. Must have columns `obs` and `time1` generated by `filter_time`.
`dat.seg.list`	A list of data associated with each animal ID where names of list elements are the ID names and tracks have already been segmented. Must have columns `obs` and `time1` generated by `filter_time`.
`theta.estim.long`	A data frame in long format where each observation (*time1*) of each track segment (*tseg*) of each animal ID (*id*) has separate rows for behavior proportion estimates per state. Columns for behavior and proportion estimates should be labeled *behavior* and *prop*, respectively. Date (in POSIXct format) should also be included as a column labeled *date*.
`behav.names`	deprecated. Now taken from the `theta.estim.long` object.

Value

A data frame of all animal IDs where columns (with names from `behav.names`) include proportions of each behavioral state per observation, as well as a column that stores the dominant behavior within a given track segment for which the observation belongs (`behav`). This is merged with the original data frame `dat.orig`, so any observations that were excluded (not at primary time interval) will show `NA` for behavior estimates.

Examples

```
#load original and segmented data
data(tracks)
data(tracks.seg)

#convert segmented dataset into list
tracks.list<- df_to_list(dat = tracks.seg, ind = "id")

#select only id, tseg, SL, and TA columns
tracks.seg2<- tracks.seg[,c("id","tseg","SL","TA")]

#summarize data by track segment
obs<- summarize_tsegs(dat = tracks.seg2, nbins = c(5,8))

#cluster data with LDA
res<- cluster_segments(dat = obs, gamma1 = 0.1, alpha = 0.1, ngibbs = 1000,
                       nburn = 500, nmaxclust = 7, ndata.types = 2)

#Extract proportions of behaviors per track segment
theta.estim<- extract_prop(res = res, ngibbs = 1000, nburn = 500, nmaxclust = 7)

#Create augmented matrix by replicating rows (tsegs) according to obs per tseg
theta.estim.long<- expand_behavior(dat = tracks.seg, theta.estim = theta.estim, obs = obs,
                                nbehav = 3, behav.names = c("Encamped","ARS","Transit"),
                                behav.order = c(1,2,3))

#Run function
dat.out<- assign_behavior(dat.orig = tracks, dat.seg.list = tracks.list,
                          theta.estim.long = theta.estim.long)
```

---

assign_tseg *Add segment numbers to observations*

Description

After breakpoints have been extracted for each animal ID, this function assigns the associated segment number to observations for each animal ID. These segments of observations will be used in the second stage of the model framework to perform mixed-membership clustering by Latent Dirichlet Allocation.

Usage

```
assign_tseg(dat, brkpts)
```

Arguments

dat	A list where each element stores the data for a unique animal ID. Each element is a data frame that contains all data associated for a given animal ID and must include a column labeled time1 that numbers each of the observations in consecutive order. This variable is automatically generated by the filter_time function during data preparation.
brkpts	A data frame of breakpoints for each animal ID (as generated by get_breakpts).

Value

A data frame that updates the original data object by including the segment number associated with each observation in relation to the extracted breakpoints.

Examples

```
#load data
data(tracks.list)

#subset only first track
tracks.list<- tracks.list[1]

#only retain id and discretized step length (SL) and turning angle (TA) columns
tracks.list2<- purrr::map(tracks.list,
                  subset,
                 select = c(id, SL, TA))


set.seed(1)

# Define model params
alpha<- 1
ngibbs<- 1000
nbins<- c(5,8)

#future::plan(future::multisession)  #run all MCMC chains in parallel
dat.res<- segment_behavior(data = tracks.list2, ngibbs = ngibbs, nbins = nbins,
                           alpha = alpha)


# Determine MAP iteration for selecting breakpoints and store breakpoints
MAP.est<- get_MAP(dat = dat.res$LML, nburn = ngibbs/2)
brkpts<- get_breakpts(dat = dat.res$brkpts, MAP.est = MAP.est)


# Assign track segments to all observations by ID
tracks.seg<- assign_tseg(dat = tracks.list, brkpts = brkpts)
```

assign_tseg_internal *Internal function that adds segment numbers to observations*

Description

After breakpoints have been extracted for each animal ID, this function assigns the associated segment number to observations for each animal ID. These segments of observations will be used in the second stage of the model framework to perform mixed-membership clustering by Latent Dirichlet Allocation.

Usage

```
assign_tseg_internal(dat, brkpts)
```

Arguments

dat	A data frame that contains all data associated for a given animal ID. Must include a column labeled time1 that numbers each of the observations in consecutive order, which is automatically generated by filter_time.
brkpts	A data frame of breakpoints for each animal ID (as generated by get_breakpts).

Value

A data frame that updates the original data object by including the segment number associated with each observation in relation to the extracted breakpoints.

behav_gibbs_sampler *Internal function that runs RJMCMC on a single animal ID*

Description

This function serves as a wrapper for samp_move by running this sampler for each iteration of the MCMC chain. It is called by segment_behavior to run the RJMCMC on all animal IDs simultaneously.

Usage

```
behav_gibbs_sampler(dat, ngibbs, nbins, alpha, breakpt, p)
```

Arguments

dat	A data frame that only contains columns for the animal IDs and for each of the discretized movement variables.
ngibbs	numeric. The total number of iterations of the MCMC chain.
nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within dat.
alpha	numeric. A single value used to specify the hyperparameter for the prior distribution. A standard value for alpha is typically 1, which corresponds with a vague prior on the Dirichlet distribution.
breakpt	numeric. A vector of breakpoints if pre-specifying where they may occur, otherwise NULL.
p	An object storing information from progressr::progessor to produce a progress bar.

Value

A list of the breakpoints, the number of breakpoints, and the log marginal likelihood at each MCMC iteration, as well as the time it took the model to finish running. This is only provided for the data of a single animal ID.

---

behav_seg_image *Internal function that transforms a vector of bin numbers to a presence-absence matrix*

Description

Transforms vectors of bin numbers into full matrices for plotting as a heatmap.

Usage

```
behav_seg_image(dat, nbins)
```

Arguments

dat	A data frame for a single animal ID that contains only columns for the ID and each of the movement variables that were analyzed by segment_behavior. The ID column must be first.
nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within data.

Value

A list where each element stores the presence-absence matrix for each of the movement variables.

cluster_obs *Cluster observations into behavioral states*

Description

This function uses a Gibbs sampler within a mixture model to estimate the optimal number of behavioral states, the state-dependent distributions, and to assign behavioral states to each observation. This model does not assume an underlying mechanistic process.

Usage

```
cluster_obs(dat, alpha, ngibbs, nmaxclust, nburn)
```

Arguments

dat	A data frame that **only** contains columns for the discretized movement variables.
alpha	numeric. A single value used to specify the hyperparameter for the prior distribution.
ngibbs	numeric. The total number of iterations of the MCMC chain.
nmaxclust	numeric. A single number indicating the maximum number of clusters to test.
nburn	numeric. The length of the burn-in phase.

Details

The mixture model analyzes all animal IDs pooled together, thus providing a population-level estimate of behavioral states.

Value

A list of model results is returned where elements include the `phi` matrix for each data stream, `theta` matrix, log likelihood estimates for each iteration of the MCMC chain `loglikel`, a list of the MAP estimates of the latent states for each observation `z.MAP`, a matrix of the whole posterior of state assignments per observation `z.posterior`, and a vector `gamma1` of estimates for the gamma hyperparameter.

Examples

```
data(tracks.list)

#convert from list to data frame
tracks.list<- dplyr::bind_rows(tracks.list)

#only retain id and discretized step length (SL) and turning angle (TA) columns
tracks<- subset(tracks.list, select = c(SL, TA))
```

```
set.seed(1)

# Define model params
alpha=0.1
ngibbs=1000
nburn=ngibbs/2
nmaxclust=7

dat.res<- cluster_obs(dat = tracks, alpha = alpha, ngibbs = ngibbs,
                            nmaxclust = nmaxclust, nburn = nburn)
```

`cluster_segments` *Cluster time segments into behavioral states*

Description

This function performs a Gibbs sampler within the Latent Dirichlet Allocation (LDA) model to estimate proportions of each behavioral state for all time segments generated by `segment_behavior`. This is the second stage of the two-stage Bayesian model that estimates proportions of behavioral states by first segmenting individual tracks into relatively homogeneous segments of movement.

Usage

```
cluster_segments(dat, gamma1, alpha, ngibbs, nmaxclust, nburn, ndata.types)
```

Arguments

`dat`	A data frame returned by `summarize_tsegs` that summarizes the counts of observations per bin and movement variable for all animal IDs.
`gamma1`	numeric. A hyperparameter for the truncated stick-breaking prior for estimating the `theta` matrix.
`alpha`	numeric. A hyperparameter for the Dirichlet distribution when estimating the `phi` matrix.
`ngibbs`	numeric. The total number of iterations of the MCMC chain.
`nmaxclust`	numeric. A single number indicating the maximum number of clusters to test.
`nburn`	numeric. The length of the burn-in phase.
`ndata.types`	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within `dat`.

Details

The LDA model analyzes all animal IDs pooled together, thereby providing population-level estimates of behavioral states.

Value

A list of model results is returned where elements include the `phi` matrix for each data stream, `theta` matrix, log likelihood estimates for each iteration of the MCMC chain `loglikel`, and matrices of the latent cluster estimates for each data stream `z.agg`.

Examples

```
#load data
data(tracks.seg)

#select only id, tseg, SL, and TA columns
tracks.seg2<- tracks.seg[,c("id","tseg","SL","TA")]

#summarize data by track segment
obs<- summarize_tsegs(dat = tracks.seg2, nbins = c(5,8))

#cluster data with LDA
res<- cluster_segments(dat = obs, gamma1 = 0.1, alpha = 0.1, ngibbs = 1000,
                       nburn = 500, nmaxclust = 7, ndata.types = 2)
```

CumSumInv *Internal function that calculates the inverted cumsum*

Description

Internal function that calculates the inverted cumsum

Usage

```
CumSumInv(ntsegm, nmaxclust, z)
```

Arguments

`ntsegm`	An integer.
`nmaxclust`	An integer.
`z`	An integer matrix.

df_to_list *Convert data frame to a list by animal ID*

Description

Converts an object of class `data.frame` to a list where each element is a separate animal ID. This function prepares the data for further analysis and when mapping other functions onto the data for separate animal IDs.

Usage

```
df_to_list(dat, ind)
```

Arguments

`dat`	A data frame containing the data for each animal ID.
`ind`	character. The name of the column storing the animal IDs.

Value

A list where each element stores the data for a separate animal ID.

Examples

```
#load data
data(tracks)

#convert to list
dat.list<- df_to_list(dat = tracks, ind = "id")
```

discrete_move_var *Discretize movement variables*

Description

Convert movement variables from continuous to discrete values for analysis by `segment_behavior`.

Usage

```
discrete_move_var(dat, lims, varIn, varOut)
```

Arguments

dat	A data frame that contains the variable(s) of interest to convert from continuous to discrete values.
lims	A list of the bin limits for each variable. Each element of the list should be a vector of real numbers.
varIn	A vector of names for the continuous variable stored as columns within dat.
varOut	A vector of names for the storage of the discrete variables returned by the function.

Value

A data frame with new columns of discretized variables as labeled by varOut.

Examples

```
#load data
data(tracks)

#subset only first track
tracks<- tracks[tracks$id == "id1",]

#calculate step lengths and turning angles
tracks<- prep_data(dat = tracks, coord.names = c("x","y"), id = "id")

#round times to nearest interval of interest (e.g. 3600 s or 1 hr)
tracks<- round_track_time(dat = tracks, id = "id", int = 3600, tol = 180, time.zone = "UTC",
                                  units = "secs")

#create list from data frame
tracks.list<- df_to_list(dat = tracks, ind = "id")

#filter observations to only 1 hr (or 3600 s)
tracks_filt.list<- filter_time(dat.list = tracks.list, int = 3600)

#define bin number and limits for turning angles and step lengths
angle.bin.lims=seq(from=-pi, to=pi, by=pi/4)  #8 bins
dist.bin.lims=quantile(tracks[tracks$dt == 3600,]$step,
                       c(0,0.25,0.50,0.75,0.90,1), na.rm=TRUE)  #5 bins


# Assign bins to observations
tracks_disc.list<- purrr::map(tracks_filt.list,
                       discrete_move_var,
                       lims = list(dist.bin.lims, angle.bin.lims),
                       varIn = c("step", "angle"),
                       varOut = c("SL", "TA"))
```

expand_behavior *Expand behavior estimates from track segments to observations*

Description

Expand behavior estimates from track segments to observations

Usage

```
expand_behavior(dat, theta.estim, obs, nbehav, behav.names, behav.order)
```

Arguments

dat	A data frame of the animal ID, track segment labels, and all other data per observation. Animal ID, date, track segment, and observation number columns must be labeled *id*, *date*, *tseg*, and *time1*, respectively.
theta.estim	A matrix (returned by extract_prop) containing the proportions of each behavioral state as separate columns for each track segment (rows).
obs	A data frame summarizing the number of observations within each bin per movement variable that is returned by summarize_tsegs.
nbehav	numeric. The number of behavioral states that will be retained in 1 to nmaxclust.
behav.names	character. A vector of names to label each state (in order).
behav.order	numeric. A vector that identifies the order in which the user would like to rearrange the behavioral states. If satisfied with order returned by the LDA model, this still must be specified.

Value

A new data frame that expands behavior proportions for each observation within all track segments, including the columns labeled *time1* and *date* from the original dat data frame.

Examples

```
#load data
data(tracks.seg)

#select only id, tseg, SL, and TA columns
tracks.seg2<- tracks.seg[,c("id","tseg","SL","TA")]

#summarize data by track segment
obs<- summarize_tsegs(dat = tracks.seg2, nbins = c(5,8))

#cluster data with LDA
res<- cluster_segments(dat = obs, gamma1 = 0.1, alpha = 0.1, ngibbs = 1000,
                       nburn = 500, nmaxclust = 7, ndata.types = 2)
```

```
#Extract proportions of behaviors per track segment
theta.estim<- extract_prop(res = res, ngibbs = 1000, nburn = 500, nmaxclust = 7)

#Create augmented matrix by replicating rows (tsegs) according to obs per tseg
theta.estim.long<- expand_behavior(dat = tracks.seg, theta.estim = theta.estim, obs = obs,
                                   nbehav = 3, behav.names = c("Encamped","ARS","Transit"),
                                   behav.order = c(1,2,3))
```

`extract_prop` *Extract behavior proportion estimates for each track segment*

Description

Calculates the mean of the posterior for the proportions of each behavior within track segments. These results can be explored to determine the optimal number of latent behavioral states.

Usage

```
extract_prop(res, ngibbs, nburn, nmaxclust)
```

Arguments

`res`	A list of results returned by `cluster_segments`. Element `theta` stores estimate for behavior proportions for all time segments.
`ngibbs`	numeric. The total number of iterations of the MCMC chain.
`nburn`	numeric. The length of the burn-in phase.
`nmaxclust`	numeric. A single number indicating the maximum number of clusters to test.

Value

A matrix that stores the proportions of each state/cluster (columns) per track segment (rows).

Examples

```
#load data
data(tracks.seg)

#select only id, tseg, SL, and TA columns
tracks.seg2<- tracks.seg[,c("id","tseg","SL","TA")]

#summarize data by track segment
obs<- summarize_tsegs(dat = tracks.seg2, nbins = c(5,8))

#cluster data with LDA
```

```
res<- cluster_segments(dat = obs, gamma1 = 0.1, alpha = 0.1, ngibbs = 1000,
                       nburn = 500, nmaxclust = 7, ndata.types = 2)

#Extract proportions of behaviors per track segment
theta.estim<- extract_prop(res = res, ngibbs = 1000, nburn = 500, nmaxclust = 7)
```

`filter_time` *Filter observations for time interval of interest*

Description

Selects observations that belong to the time interval of interest and removes all others. This function also removes entire IDs from the dataset when there is one or fewer observations at this time interval. This function works closely with `round_track_time` to only retain observations sampled at a regular time interval, which is important for analyzing step lengths and turning angles. Column storing the time intervals must be labeled `dt`.

Usage

```
filter_time(dat.list, int)
```

Arguments

`dat.list`	A list of data associated with each animal ID where names of list elements are the ID names.
`int`	numeric. The time interval of interest.

Value

A list where observations for each animal ID (element) has been filtered for `int`. Two columns (`obs` and `time1`) are added for each list element (ID), which store the original observation number before filtering and the new observation number after filtering, respectively.

Examples

```
#load data
data(tracks)

#subset only first track
tracks<- tracks[tracks$id == "id1",]

#calculate step lengths and turning angles
tracks<- prep_data(dat = tracks, coord.names = c("x","y"), id = "id")

#round times to nearest interval of interest (e.g. 3600 s or 1 hr)
tracks<- round_track_time(dat = tracks, id = "id", int = 3600, tol = 180, time.zone = "UTC",
```

```
                                units = "secs")

#create list from data frame
tracks.list<- df_to_list(dat = tracks, ind = "id")

#filter observations to only 1 hr (or 3600 s)
tracks_filt.list<- filter_time(dat.list = tracks.list, int = 3600)
```

`find_breaks` *Find changes for integer variable*

Description

Identify changes within a discrete variable. These values can be used to pre-specify breakpoints within the segmentation model using `segment_behavior`.

Usage

```
find_breaks(dat, ind)
```

Arguments

`dat`	A data frame containing the data for each animal ID.
`ind`	character. The name of the column storing the discrete variable of interest.

Value

A vector of breakpoints is returned based on the data provided. If wishing to identify separate breakpoints per animal ID, this function should be mapped onto a list generated by `df_to_list`.

Examples

```
#simuluate data
var<- sample(1:3, size = 50, replace = TRUE)
var<- rep(var, each = 20)
id<- rep(1:10, each = 100)

#create data frame
dat<- data.frame(id, var)

#create list
dat.list<- df_to_list(dat = dat, ind = "id")

#run function using purrr::map()
breaks<- purrr::map(dat.list, ~find_breaks(dat = ., ind = "var"))

#or with lapply()
breaks1<- lapply(dat.list, find_breaks, ind = "var")
```

get.llk.mixmod — *Internal function to calculate the log-likelihood for iteration of mixture model*

Description

Calculates the log-likelihood of the mixture model based on estimates for *theta* and *phi*.

Usage

```
get.llk.mixmod(phi, theta, ndata.types, dat, nobs, nmaxclust)
```

Arguments

phi	A list of proportion estimates that characterize distributions (bins) for each data stream and possible behavioral state.
theta	numeric. A vector of values that sum to one.
ndata.types	numeric. The number of data streams being analyzed.
dat	A data frame containing only columns of the discretized data streams for all observations.
nobs	numeric. The total number of rows in the dataset.
nmaxclust	numeric. A single number indicating the maximum number of clusters to test.

Value

A numeric value of the log-likelihood based upon the current values for *phi* and *theta*.

get.theta — *Internal function to calculate theta parameter*

Description

Calculates values of *theta* matrix within Gibbs sampler. Not for calling directly by users.

Usage

```
get.theta(v, nmaxclust, ntsegm)
```

Arguments

v	A matrix returned by sample.v
nmaxclust	numeric. A single number indicating the maximum number of clusters to test.
ntsegm	numeric. The total number of time segments from all animal IDs.

Value

A matrix of proportion estimates that represent proportions of different behavioral states per time segment.

`get_behav_hist` *Extract bin estimates from Latent Dirichlet Allocation or mixture model*

Description

Pulls model results for the estimates of bin proportions per movement variable from the posterior distribution. This can be used for visualization of movement variable distribution for each behavior estimated.

Usage

```
get_behav_hist(dat, nburn, ngibbs, nmaxclust, var.names)
```

Arguments

`dat`	The list object returned by the LDA model (`cluster_segments`) or mixture model (`cluster_obs`). Used for extracting the element *phi*.
`nburn`	numeric. The length of the burn-in phase.
`ngibbs`	numeric. The total number of iterations of the MCMC chain.
`nmaxclust`	numeric. The maximum number of clusters on which to attribute behaviors.
`var.names`	character. A vector of names used for each of the movement variables. Must be in the same order as were listed within the data frame returned by `summarize_tsegs` (if running LDA model).

Value

A data frame that contains columns for bin number, behavioral state, proportion represented by a given bin, and movement variable name. This is displayed in a long format, which is easier to visualize using `ggplot2`.

Examples

```
#load data
data(tracks.seg)

#select only id, tseg, SL, and TA columns
tracks.seg2<- tracks.seg[,c("id","tseg","SL","TA")]

#summarize data by track segment
obs<- summarize_tsegs(dat = tracks.seg2, nbins = c(5,8))
```

```
#cluster data with LDA
res<- cluster_segments(dat = obs, gamma1 = 0.1, alpha = 0.1, ngibbs = 1000,
                       nburn = 500, nmaxclust = 7, ndata.types = 2)

#Extract proportions of behaviors per track segment
theta.estim<- extract_prop(res = res, ngibbs = 1000, nburn = 500, nmaxclust = 7)

#run function for clustered segments
behav.res<- get_behav_hist(dat = res, nburn = 500, ngibbs = 1000, nmaxclust = 7,
                           var.names = c("Step Length","Turning Angle"))
```

`get_breakpts` *Extract breakpoints for each animal ID*

Description

Extract breakpoints for each animal ID

Usage

```
get_breakpts(dat, MAP.est)
```

Arguments

`dat`	A list of lists where animal IDs are separated as well as the breakpoints estimated for each iteration of the MCMC chain. This is stored within `breakpts` of model results returned after running `segment_behavior`.
`MAP.est`	numeric. A vector of values at which the maximum a posteriori (MAP) estimate was identified for each of the animal IDs as returned by `get_MAP`. These must be in the same order as the data for the IDs supplied to `segment_behavior()`.

Value

A data frame where breakpoints are returned per animal ID within each row. For animal IDs that have fewer breakpoints than the maximum number that were estimated, `NA` values are used as place holders for these breakpoints that do not exist.

Examples

```
#load data
data(tracks.list)

#subset only first track
tracks.list<- tracks.list[1]

#only retain id and discretized step length (SL) and turning angle (TA) columns
tracks.list2<- purrr::map(tracks.list,
```

```
                    subset,
                   select = c(id, SL, TA))


set.seed(1)

# Define model params
alpha<- 1
ngibbs<- 1000
nbins<- c(5,8)

#future::plan(future::multisession)  #run all MCMC chains in parallel
dat.res<- segment_behavior(data = tracks.list2, ngibbs = ngibbs, nbins = nbins,
                           alpha = alpha)


# Determine MAP iteration for selecting breakpoints and store breakpoints
MAP.est<- get_MAP(dat = dat.res$LML, nburn = ngibbs/2)
brkpts<- get_breakpts(dat = dat.res$brkpts, MAP.est = MAP.est)
```

`get_MAP` *Find the maximum a posteriori (MAP) estimate of the MCMC chain*

Description

Identify the MCMC iteration that holds the MAP estimate. This will be used to inform `get_breakpts` as to which breakpoints should be retained on which to assign track segments to the observations of each animal ID.

Usage

```
get_MAP(dat, nburn)
```

Arguments

`dat`	A data frame where each row holds the log marginal likelihood values at each iteration of the MCMC chain.
`nburn`	numeric. The size of the burn-in phase after which the MAP estimate will be identified.

Value

A numeric vector of iterations at which the MAP estimate was found for each animal ID.

Examples

```
#load data
data(tracks.list)

#subset only first track
tracks.list<- tracks.list[1]

#only retain id and discretized step length (SL) and turning angle (TA) columns
tracks.list2<- purrr::map(tracks.list,
                  subset,
                 select = c(id, SL, TA))


set.seed(1)

# Define model params
alpha<- 1
ngibbs<- 1000
nbins<- c(5,8)

#future::plan(future::multisession)  #run all MCMC chains in parallel
dat.res<- segment_behavior(data = tracks.list2, ngibbs = ngibbs, nbins = nbins,
                           alpha = alpha)


# Determine MAP iteration for selecting breakpoints and store breakpoints
MAP.est<- get_MAP(dat = dat.res$LML, nburn = ngibbs/2)
```

`get_summary_stats` *Internal function that calculates the sufficient statistics for the segmentation model*

Description

An internal function that calculates the sufficient statistics to be used within the reversible-jump MCMC Gibbs sampler called by `link{samp_move}`.

Usage

```
get_summary_stats(breakpt, dat, max.time, nbins, ndata.types)
```

Arguments

`breakpt`	numeric. A vector of breakpoints.
`dat`	A matrix that only contains columns storing discretized data for each of the movement variables.
`max.time`	numeric. The number of of the last observation of `dat`.

nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within dat.
ndata.types	numeric. The length of nbins.

Value

Returns the sufficient statistics associated with the provided breakpoints for a given animal ID.

`insert_NAs` *Insert NA gaps to regularize a time series*

Description

Insert NA gaps to regularize a time series

Usage

```
insert_NAs(data, int, units)
```

Arguments

data	A data frame that minimally contains columns for animal ID, date, and time step. These must be labeled id, date, and dt, respectively, where date is of class POSIXct.
int	integer. An integer that characterizes the desired interval on which to insert new rows.
units	character. The units of the selected time interval int, which can be selected from one of "secs", "mins", "hours", "days", or "weeks".

Value

A data frame where new rows have been inserted to regularize the date column. This results in values provided for id, date, and dt while inserting NAs for all other columns. Additionally, observations with duplicate date-times are removed.

Examples

```
#load data
data(tracks)

#remove rows to show how function works (create irregular time series)
set.seed(1)
ind<- sort(sample(2:15003, 500))

tracks.red<- tracks[-ind,]

#calculate step lengths, turning angles, net-squared displacement, and time steps
tracks.red<- prep_data(dat = tracks.red, coord.names = c("x","y"), id = "id")
```

```
#round times to nearest interval
tracks.red<- round_track_time(dat = tracks.red, id = "id", int = c(3600, 7200, 10800, 14400),
                              tol = 300, units = "secs")

#insert NA gaps
dat.out<- insert_NAs(tracks.red, int = 3600, units = "secs")
```

`log_marg_likel` *Internal function that calculates the log marginal likelihood of each model being compared*

Description

An internal function that is used to calculate the log marginal likelihood of models for the current and proposed sets of breakpoints. Called within `samp_move`.

Usage

```
log_marg_likel(alpha, summary.stats, nbins, ndata.types)
```

Arguments

`alpha`	numeric. A single value used to specify the hyperparameter for the prior distribution. A standard value for `alpha` is typically 1, which corresponds with a vague prior on the Dirichlet distribution.
`summary.stats`	A matrix of sufficient statistics returned from `get_summary_stats`.
`nbins`	numeric. A vector of the number of bins used to discretize each movement variable.
`ndata.types`	numeric. The length of `nbins`.

Value

The log marginal likelihood is calculated for a model with a given set of breakpoints and the discretized data.

plot_breakpoints *Plot breakpoints over a time series of each movement variable*

Description

Visualize the breakpoints estimated by the segmentation model as they relate to either the original (continuous) or discretized data. These plots assist in determining whether too many or too few breakpoints were estimated as well as whether the user needs to redefine how they discretized their data before analysis.

Usage

```
plot_breakpoints(data, as_date = FALSE, var_names, var_labels = NULL, brkpts)
```

Arguments

data	A list where each element stores a data frame for a given animal ID. Each of these data frames contains columns for the ID, date or time1 generated by filter_time, as well as each of the movement variables analyzed by segment_behavior.
as_date	logical. If TRUE, plots breakpoints and data streams over the date. By default, this is set to FALSE.
var_names	A vector of the column names for the movement variables to be plotted over time.
var_labels	A vector of the labels to be plotted on the y-axis for each movement variable. Set to NULL by default.
brkpts	A data frame that contains the breakpoints associated with each animal ID. This data frame is returned by get_breakpts.

Value

A line plot per animal ID for each movement variable showing how the estimated breakpoints relate to the underlying data. Depending on the user input for var_names, this may either be on the scale of the original continuous data or the discretized data.

Examples

```
#load data
data(tracks.list)

#subset only first track
tracks.list<- tracks.list[1]

#only retain id and discretized step length (SL) and turning angle (TA) columns
tracks.list2<- purrr::map(tracks.list,
                  subset,
                 select = c(id, SL, TA))
```

```
set.seed(1)

# Define model params
alpha<- 1
ngibbs<- 1000
nbins<- c(5,8)

#future::plan(future::multisession)  #run all MCMC chains in parallel
dat.res<- segment_behavior(data = tracks.list2, ngibbs = ngibbs, nbins = nbins,
                           alpha = alpha)


# Determine MAP iteration for selecting breakpoints and store breakpoints
MAP.est<- get_MAP(dat = dat.res$LML, nburn = ngibbs/2)
brkpts<- get_breakpts(dat = dat.res$brkpts, MAP.est = MAP.est)


#run function
plot_breakpoints(data = tracks.list, as_date = FALSE, var_names = c("step","angle"),
    var_labels = c("Step Length (m)", "Turning Angle (rad)"), brkpts = brkpts)
```

`plot_breakpoints_behav` *Internal function for plotting breakpoints over each of the data streams*

Description

An internal function for plotting the results of the segmentation model.

Usage

```
plot_breakpoints_behav(data, as_date, var_names, var_labels, brkpts)
```

Arguments

`data`	A data frame for a single animal ID that contains columns for the ID, date or time variable, and each of the movement variables that were analyzed by `segment_behavior`. Data streams can be in continuous or discrete form.
`as_date`	logical. If `TRUE`, plots breakpoints and data streams over the date. By default, this is set to `FALSE`.
`var_names`	A vector of the column names for the movement variables to be plotted over time.
`var_labels`	A vector of the labels to be plotted on the y-axis for each movement variable. Set to `NULL` by default.
`brkpts`	A data frame that contains the breakpoints associated with each animal ID. This data frame is returned by `get_breakpts`.

Value

A line plot for each movement variable showing how the estimated breakpoints relate to the underlying data. Depending on the user input for `var_names`, this may either be on the scale of the original continuous data or the discretized data.

`prep_data` *Calculate step lengths, turning angles, net-squared displacement, and time steps*

Description

Calculates step lengths, turning angles, and net-squared displacement based on coordinates for each animal ID and calculates time steps based on the date-time. Provides a self-contained method to calculate these variables without needing to rely on other R packages (e.g., `adehabitatLT`). However, functions from other packages can also be used to perform this step in data preparation.

Usage

```
prep_data(dat, coord.names, id)
```

Arguments

`dat`	A data frame that contains a column for animal IDs, the columns associated with the x and y coordinates, and a column for the date. For easier interpretation of the model results, it is recommended that coordinates be stored in a UTM projection (meters) as opposed to unprojected in decimal degrees (map units). Date-time should be of class `POSIXct` and be labeled `date` within the data frame.
`coord.names`	character. A vector of the column names under which the coordinates are stored. The name for the x coordinate should be listed first and the name for the y coordinate second.
`id`	character. The name of the column storing the animal IDs.

Value

A data frame where all original data are returned and new columns are added for step length (`step`), turning angle (`angle`), net-squared displacement (`NSD`), and time step (`dt`). Names for coordinates are changed to `x` and `y`. Units for `step` and `NSD` depend on the projection of the coordinates, `angle` is returned in radians, and `dt` is returned in seconds.

Examples

```
#load data
data(tracks)

#subset only first track
tracks<- tracks[tracks$id == "id1",]
```

```
#calculate step lengths and turning angles
tracks<- prep_data(dat = tracks, coord.names = c("x","y"), id = "id")
```

`prep_data_internal` *Internal function to calculate step lengths, turning angles, and time steps*

Description

An internal function that calculates step lengths, turning angles, and time steps for a given animal ID.

Usage

```
prep_data_internal(dat, coord.names)
```

Arguments

`dat`	A data frame that contains the columns associated with the x and y coordinates as well as the date-time. For easier interpretation of the model results, it is recommended that coordinates be stored after UTM projection (meters) as opposed to unprojected in decimal degrees (map units). Date-time should be of class `POSIXct` and be labeled `date` within the data frame.
`coord.names`	character. A vector of the column names under which the coordinates are stored. The name for the x coordinate should be listed first and the name for the y coordinate second.

Value

A data frame where all original data are returned and new columns are added for step length (`step`), turning angle (`angle`), net-squared displacement (`NSD`), and time step (`dt`).

`rmultinom1` *Internal function that samples z's from a categorical distribution*

Description

Internal function that samples z's from a categorical distribution

Usage

```
rmultinom1(prob, randu)
```

Arguments

`prob`	A numeric matrix.
`randu`	A numeric vector.

rmultinom2 *Internal function that samples z's from a multinomial distribution*

Description

Internal function that samples z's from a multinomial distribution

Usage

```
rmultinom2(prob, n, randu, nmaxclust)
```

Arguments

prob	A numeric vector.
n	An integer.
randu	A numeric vector.
nmaxclust	An integer.

round_track_time *Round time to nearest interval*

Description

Rounds sampling intervals that are close, but not exactly the time interval of interest (e.g., 240 s instead of 300 s). This can be performed on multiple time intervals, but only using a single tolerance value. This function prepares the data to be analyzed by `segment_behavior`, which requires that all time intervals exactly match the primary time interval when analyzing step lengths and turning angles. Columns storing the time intervals and dates must be labeled `dt` and `date`, respectively, where dates are of class `POSIXct`.

Usage

```
round_track_time(dat, id, int, tol, time.zone = "UTC", units)
```

Arguments

dat	A data frame that contains the sampling interval of the observations.
id	character. The name of the column storing the animal IDs.
int	numeric. A vector of the time interval(s) of on which to perform rounding.
tol	numeric. A single tolerance value on which to round any `int` that were specified.
time.zone	character. Specify the time zone for which the date-times were recorded. Set to UTC by default. Refer to `base::OlsonNames` to view all possible time zones.
units	character. The units of the selected time interval `int`, which can be selected from one of "secs", "mins", "hours", "days", or "weeks".

Value

A data frame where dt and date are both adjusted based upon the rounding of time intervals according to the specified tolerance.

Examples

```
#load data
data(tracks)

#subset only first track
tracks<- tracks[tracks$id == "id1",]

#calculate step lengths and turning angles
tracks<- prep_data(dat = tracks, coord.names = c("x","y"), id = "id")

#round times to nearest interval of interest (e.g. 3600 s or 1 hr)
tracks<- round_track_time(dat = tracks, id = "id", int = 3600, tol = 180, time.zone = "UTC",
                          units = "secs")
```

---

sample.gamma.mixmod *Internal function to sample the gamma hyperparameter*

Description

Internal function to sample the gamma hyperparameter

Usage

```
sample.gamma.mixmod(v, ngroup, gamma.possib)
```

Arguments

v	numeric. A vector of proportions for each of the possible clusters.
ngroup	numeric. The total number of possible clusters.
gamma.possib	numeric. A vector of possible values that gamma can take ranging between 0.1 and 1.

Value

A single numeric value for gamma that falls within gamma.possib for calculation of the log-likelihood.

sample.phi — *Internal function to sample bin estimates for each movement variable*

Description

Estimates values of *phi* matrix for use in characterizing distributions of the movement variables. Not for calling directly by users.

Usage

```
sample.phi(z.agg, alpha, nmaxclust, nbins, ndata.types)
```

Arguments

z.agg	A list of latent cluster estimates provided by sample.z.
alpha	numeric. A hyperparameter for the Dirichlet distribution.
nmaxclust	numeric. A single number indicating the maximum number of clusters to test.
nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within y.
ndata.types	numeric. The number of data streams being analyzed.

Value

A matrix of proportion estimates that characterize distributions (bins) for each movement variable and possible behavioral state.

sample.phi.mixmod — *Internal function to sample bin estimates for each movement variable*

Description

Estimates values of *phi* matrix for use in characterizing distributions of the movement variables. Not for calling directly by users.

Usage

```
sample.phi.mixmod(alpha, nmaxclust, nbins, ndata.types, nmat)
```

Arguments

alpha	numeric. A hyperparameter for the Dirichlet distribution.
nmaxclust	numeric. A single number indicating the maximum number of clusters to test.
nbins	numeric. A vector of the number of bins used to discretize each data stream. These must be in the same order as the columns within dat.
ndata.types	numeric. The number of data streams being analyzed.
nmat	A list based on SummarizeDat C++ function to help with multinomial draws.

Value

A list of proportion estimates that characterize distributions (bins) for each data stream and possible behavioral state.

`sample.v` *Internal function to sample parameter for truncated stick-breaking prior*

Description

This function samples the latent *v* parameter within the Gibbs sampler. Calls on the `CumSumInv` function written in C++. Not for calling directly by users.

Usage

```
sample.v(z.agg, gamma1, ntsegm, ndata.types, nmaxclust)
```

Arguments

`z.agg`	A list of latent cluster estimates provided by `sample.z`.
`gamma1`	numeric. Hyperparameter for the truncated stick-breaking prior.
`ntsegm`	numeric. The total number of time segments from all animal IDs.
`ndata.types`	numeric. The number of data streams being analyzed.
`nmaxclust`	numeric. A single number indicating the maximum number of clusters to test.

Value

A matrix with estimates for *v* for each of the number of time segments and possible states.

`sample.v.mixmod` *Internal function to sample parameter for truncated stick-breaking prior*

Description

This function samples the latent *v* parameter within the Gibbs sampler. Not for calling directly by users.

Usage

```
sample.v.mixmod(z, gamma1, nmaxclust)
```

Arguments

z	A vector of latent cluster estimates provided by sample.z.mixmod.
gamma1	numeric. Hyperparameter for the truncated stick-breaking prior.
nmaxclust	numeric. A single number indicating the maximum number of clusters to test.

Value

A list with estimates for *v* and *theta* for each of the possible states.

---

sample.z *Internal function to sample latent clusters*

Description

This function samples the latent *z* parameter within the Gibbs sampler. Calls on the SampleZAgg function written in C++. Not for calling directly by users.

Usage

```
sample.z(ntsegm, nbins, y, nmaxclust, phi, ltheta, zeroes, ndata.types)
```

Arguments

ntsegm	numeric. The total number of time segments from all animal IDs.
nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within y.
y	A list where each element stores separate aggregated count data per bin per time segment for each movement variable being analyzed. These are stored as matrices.
nmaxclust	numeric. A single number indicating the maximum number of clusters to test.
phi	A list where each element stores separate proportions per bin per time segment for each movement variable.
ltheta	A matrix storing the log-transformed values from the theta parameter.
zeroes	A list of arrays that contain only zero values which are three dimensional (ntsegm,nbins[i],nmaxclust).
ndata.types	numeric. The number of data streams being analyzed.

Value

A list with estimates for *z* where the number of elements is equal to the number of movement variables.

sample.z.mixmod *Internal function to sample latent clusters (for observations)*

Description

This function samples the latent z parameter within the Gibbs sampler. Calls on the `rmultinom1` function written in C++. Not for calling directly by users.

Usage

```
sample.z.mixmod(nobs, nmaxclust, dat, ltheta, lphi, ndata.types)
```

Arguments

`nobs`	numeric. The total number of rows in the dataset.
`nmaxclust`	numeric. A single number indicating the maximum number of clusters to test.
`dat`	A data frame containing only columns of the discretized data streams for all observations.
`ltheta`	numeric. A vector of log-transformed estimates for parameter theta.
`lphi`	A list containing log-transformed estimates for each data stream of the phi parameter.
`ndata.types`	numeric. The number of data streams being analyzed.

Value

A vector with estimates for z for each observation within `dat`.

SampleZAgg *Internal function that samples z1 aggregate*

Description

Internal function that samples z1 aggregate

Usage

```
SampleZAgg(ntsegm, b1, y1, nmaxclust, lphi1, ltheta, zeroes)
```

Arguments

ntsegm	An integer.
b1	An integer.
y1	An integer matrix.
nmaxclust	An integer.
lphi1	A numeric matrix.
ltheta	A numeric matrix.
zeroes	A numeric vector.

---

samp_move *Internal function for the Gibbs sampler within the reversible-jump MCMC algorithm*

Description

This is RJMCMC algorithm that drives the proposal and selection of breakpoints for the data based on the difference in log marginal likelihood. This function is called within behav_gibbs_sampler.

Usage

```
samp_move(breakpt, max.time, dat, alpha, nbins, ndata.types)
```

Arguments

breakpt	numeric. A vector of breakpoints.
max.time	numeric. The number of of the last observation of dat.
dat	A matrix that only contains columns storing discretized data for each of the movement variables used within get_summary_stats.
alpha	numeric. A single value used to specify the hyperparameter for the prior distribution. A standard value for alpha is typically 1, which corresponds with a vague prior on the Dirichlet distribution.
nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within dat.
ndata.types	numeric. The length of nbins.

Value

The breakpoints and log marginal likelihood are retained from the selected model from the Gibbs sampler and returned as elements of a list. This is performed for each iteration of the MCMC algorithm.

segment_behavior *Segmentation model to estimate breakpoints*

Description

This function performs the reversible-jump MCMC algorithm using a Gibbs sampler, which estimates the breakpoints of the movement variables for each of the animal IDs. This is the first stage of the two-stage Bayesian model that estimates proportions of behavioral states by first segmenting individual tracks into relatively homogeneous segments of movement.

Usage

```
segment_behavior(
  data,
  ngibbs,
  nbins,
  alpha,
  breakpt = purrr::map(names(data), ~NULL)
)
```

Arguments

data	A list where each element stores the data for a separate animal ID. List elements are data frames that only contain columns for the animal ID and for each of the discretized movement variables.
ngibbs	numeric. The total number of iterations of the MCMC chain.
nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within data.
alpha	numeric. A single value used to specify the hyperparameter for the prior distribution. A standard value for alpha is typically 1, which corresponds with a vague prior on the Dirichlet distribution.
breakpt	A list where each element stores a vector of breakpoints if pre-specifying where they may occur for each animal ID. By default this is set to NULL.

Details

This model is run in parallel using the future package. To ensure that the model is run in parallel, the plan must be used with future::multisession as the argument for most operating systems. Otherwise, model will run sequentially by default if this is not set before running segment_behavior.

Value

A list of model results is returned where elements include the breakpoints, number of breakpoints, and log marginal likelihood at each iteration of the MCMC chain for all animal IDs. The time it took the model to finish running for each animal ID are also stored and returned.

Examples

```
#load data
data(tracks.list)

#subset only first track
tracks.list<- tracks.list[1]

#only retain id and discretized step length (SL) and turning angle (TA) columns
tracks.list2<- purrr::map(tracks.list,
                  subset,
                 select = c(id, SL, TA))


set.seed(1)

# Define model params
alpha<- 1
ngibbs<- 1000
nbins<- c(5,8)

future::plan(future::multisession, workers = 3)  #run all MCMC chains in parallel

dat.res<- segment_behavior(data = tracks.list2, ngibbs = ngibbs, nbins = nbins,
                           alpha = alpha)


future::plan(future::sequential)  #return to single core
```

`shiny_tracks` *Dynamically explore tracks within Shiny app*

Description

This Shiny application allows for the exploration of animal movement patterns. Options are available to interactively filter the plotted tracks by a selected time period of a given variable, which is then displayed on an interactive map. Additionally, a data table is shown with options to filter and export this table once satisfied.

Usage

```
shiny_tracks(data, epsg)
```

Arguments

`data`	A data frame that must contain columns labeled `id`, `x`, `y`,`date`, but can include any other variables of interest.
`epsg`	numeric. The coordinate reference system (CRS) as an EPSG code or a PROJ string.

Details

Currently, the time series plot shown for the exploration of individual tracks cannot display variables of class `character` or `factor`. Therefore, these should be changed to numeric values if they are to be plotted.

If the data are stored as longitude and latitude (i.e., WGS84), the EPSG code is 4326. All other codes will need to be looked up if they are not already known.

Examples

```
## Not run:
#load data
data(tracks)

#run Shiny app
shiny_tracks(data = tracks, epsg = 32617)


## End(Not run)
```

StoreZ *This function helps store z from all iterations after burn in*

Description

This function helps store z from all iterations after burn in

Usage

```
StoreZ(z, store_z, nobs)
```

Arguments

`z`	An integer vector.
`store_z`	An integer matrix.
`nobs`	An integer.

summarize1 *Internal function that summarizes bin distributions of track segments*

Description

Internal function that summarizes bin distributions of track segments

Usage

```
summarize1(VecVals, Breakpts, nobs, nbins, nbreak)
```

Arguments

VecVals	A vector of bin values.
Breakpts	A vector if breakpoints.
nobs	The number of observations.
nbins	The number of bins for a given data stream.
nbreak	The number of estimated breakpoints.

SummarizeDat *Internal function that generates nmat matrix to help with multinomial draws*

Description

Internal function that generates nmat matrix to help with multinomial draws

Usage

```
SummarizeDat(z, dat, ncateg, nbehav, nobs)
```

Arguments

z	An integer vector.
dat	An integer vector.
ncateg	An integer.
nbehav	An integer.
nobs	An integer.

summarize_tsegs *Summarize observations within bins per track segment*

Description

Prepares the data that has already been segmented for clustering by Latent Dirichlet Allocation. This function summarizes the counts observed per movement variable bin within each track segment per animal ID.

Usage

```
summarize_tsegs(dat, nbins)
```

Arguments

dat	A data frame of **only** the animal ID, track segment number, and the discretized data for each movement variable. Animal ID and time segment must be the first two columns of this data frame. This should be a simplified form of the output from assign_tseg.
nbins	numeric. A vector of the number of bins used to discretize each movement variable. These must be in the same order as the columns within dat.

Value

A new data frame that contains the animal ID, track segment number, and the counts per bin for each movement variable. The names for each of these bins are labeled according to the order in which the variables were provided to summarize_tsegs.

Examples

```
#load data
data(tracks.seg)

#select only id, tseg, SL, and TA columns
tracks.seg2<- tracks.seg[,c("id","tseg","SL","TA")]

#run function
obs<- summarize_tsegs(dat = tracks.seg2, nbins = c(5,8))
```

traceplot *View trace-plots of output from Bayesian segmentation model*

Description

Visualize trace-plots of the number of breakpoints estimated by the model as well as the log marginal likelihood (LML) for each animal ID.

Usage

```
traceplot(data, type)
```

Arguments

data	A list of model results that is returned as output from segment_behavior.
type	character. The type of data that are being plotted from the Bayesian segmentation model results. Takes either 'nbrks' for the number of breakpoints or 'LML' for the log marginal likelihood.

Value

Trace-plots for the number of breakpoints or the log marginal likelihood are displayed for each of the animal IDs that were analyzed by the segmentation model.

Examples

```
#load data
data(tracks.list)

#only retain id and discretized step length (SL) and turning angle (TA) columns
tracks.list2<- purrr::map(tracks.list,
                  subset,
                 select = c(id, SL, TA))


set.seed(1)

# Define model params
alpha<- 1
ngibbs<- 1000
nbins<- c(5,8)

future::plan(future::multisession, workers = 3)  #run all MCMC chains in parallel

dat.res<- segment_behavior(data = tracks.list2, ngibbs = ngibbs, nbins = nbins,
                               alpha = alpha)

future::plan(future::sequential)  #return to single core
```

```
#run function
traceplot(data = dat.res, type = "nbrks")
traceplot(data = dat.res, type = "LML")
```

`tracks` *Simulated set of three tracks.*

Description

A dataset containing the IDs as well as x and y coordinates for three tracks of 5001 observations each (15,003 in total).

Usage

```
tracks
```

Format

A data frame with 15003 rows and 4 variables:

id ID for each simulated track

date date, recorded as datetime

x x coordinate of tracks

y y coordinate of tracks

`tracks.list` *Tracks discretized and prepared for segmentation.*

Description

A dataset containing the prepared data after discretizing step lengths and turning angles, as well as filtering observations at the primary time step.

Usage

```
tracks.list
```

Format

A list with three elements, each containing a data frame with ~4700 rows and 11 variables:

id ID for each simulated track

date date, recorded as datetime

x x coordinate of tracks

y y coordinate of tracks

step the step length calculated as the distance between successive locations measured in units

angle the relative turning angle measured in radians

dt the time step or sampling interval between datetimes of successive observations

obs the ordered number of observations per ID before filtering for the primary time step

time1 the ordered number of observations per ID after filtering for the primary time step

SL discretized step lengths, separated into five bins

TA discretized turning angles, separated into eight bins

---

tracks.seg *Segmented tracks for all IDs.*

Description

A dataset containing the filtered track data with time segments assigned to all observations on an individual basis.

Usage

```
tracks.seg
```

Format

A data frame with 14096 rows and 12 variables:

id ID for each simulated track

date date, recorded as datetime

x x coordinate of tracks

y y coordinate of tracks

step the step length calculated as the distance between successive locations measured in units

angle the relative turning angle measured in radians

dt the time step or sampling interval between datetimes of successive observations

obs the ordered number of observations per ID before filtering for the primary time step

time1 the ordered number of observations per ID after filtering for the primary time step

SL discretized step lengths, separated into five bins

TA discretized turning angles, separated into eight bins

tseg time segment assigned to a given set of observations per ID

Index