Package ‘actel’

October 12, 2022

Title Acoustic Telemetry Data Analysis

Version 1.2.1

Description Designed for studies where animals tagged with acoustic tags are expected to move through receiver arrays. This package combines the advantages of automatic sorting and checking of animal movements with the possibility for user intervention on tags that deviate from expected behaviour. The three analysis functions (explore(), migration() and residency()) allow the users to analyse their data in a systematic way, making it easy to compare results from different studies.

CJS calculations are based on Perry et al. (2012) [https://www.researchgate.net/publication/256443823_Using_mark-recapture_models_to_estimate_survival_from_telemetry_data].

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Language en-GB

Depends R (>= 3.6)

Imports circular, data.table, DiagrammeR, DiagrammeRsvg, fasttime, ggplot2, svglite, graphics, knitr, readr, reshape2, rmarkdown, rsvg, scales, stats, stringr, stringi, grDevices, utils

Suggests raster, gdistance, sp, tools, rgdal, testthat, RGtk2, gWidgets2RGtk2, gWidgets2

VignetteBuilder knitr

URL https://github.com/hugomflavio/actel

BugReports https://github.com/hugomflavio/actel/issues

NeedsCompilation no

Author Hugo Flávio [aut, cre] (<https://orcid.org/0000-0002-5174-1197>)

Maintainer Hugo Flávio <hflavio@wlu.ca>

Repository CRAN

Date/Publication 2021-01-05 21:40:02 UTC
R topics documented:

advEfficiency .......................................................... 2
calculateMatrix .......................................................... 4
createWorkspace ............................................................ 5
dataToList ................................................................. 6
distancesMatrix ............................................................ 7
dualArrayCJS ............................................................... 9
emptyMatrix ............................................................... 10
exampleWorkspace .......................................................... 10
explore ...................................................................... 11
extractCodeSpaces .......................................................... 14
extractSignals .............................................................. 15
getSpeeds ................................................................. 16
getTimes ................................................................. 17
loadShape ................................................................. 18
loadSpatial ............................................................... 20
migration ................................................................. 21
plotArray ................................................................. 26
plotDetections ............................................................. 27
plotLive ................................................................. 28
plotMoves ............................................................... 30
plotRatios ............................................................... 31
plotResidency ............................................................ 32
plotSensors .............................................................. 33
plotTimes ............................................................... 34
preload ................................................................. 36
readDot ................................................................. 38
recoverLog .............................................................. 39
residency ............................................................... 40
simpleCJS ............................................................... 44
stationName ............................................................. 45
timesToCircular .......................................................... 46
transitionLayer .......................................................... 47

Index ................................................................. 49

---

advEfficiency .......................................................... 2

Calculate beta estimations for efficiency

---

Description

advEfficiency estimates efficiency ranges by fitting a beta distribution with parameters $\alpha =$ number of detected tags and $\beta =$ number of missed tags. The desired quantiles (argument q) are then calculated from distribution. Plots are also drawn showing the distribution, the median point (dashed red line) and the range between the lowest and largest quantile requested (red shaded section).
advEfficiency

Usage

advEfficiency(
  x,
  labels = NULL,
  q = c(0.025, 0.5, 0.975),
  force.grid = NULL,
  paired = TRUE,
  title = ""
)

Arguments

x        An efficiency object from actel (overall.CJS, intra.array.CJS[[...]] or efficiency objects)
labels   a vector of strings to substitute default plot labels
q         The quantile values to be calculated. Defaults to c(0.025, 0.5, 0.975) (i.e. median and 95% CI)
force.grid A vector of format c(nrow, ncol) that allows the user to define the number of rows and columns to distribute the plots in.
paired    Logical: For efficiency derived from residency analyses, should min. and max. estimates for an array be displayed next to each other?
title     A title for the plot (feeds into title parameter of ggplot's labs function).

Details

Examples for inclusion in a paper:

1. If advEfficiency was run on an overall.CJS object (i.e. migration analysis):

   "Array efficiency was estimated by fitting a beta distribution (\(\alpha\) = number of tags detected subsequently and at the array, \(\beta\) = number of tags detected subsequently but not at the array) and calculating the median estimated efficiency value using the R package actel [citation]."

2. If advEfficiency was run on an efficiency object (i.e. residency analysis):

   • If you are using maximum efficiency estimates:
     "Array efficiency was estimated by fitting a beta distribution (\(\alpha\) = number of events recorded by the array, \(\beta\) = number of events known to have been missed by the array). and calculating the median estimated efficiency value using the R package actel [citation]."

   • If you are using minimum efficiency estimates:
     "Array efficiency was estimated by fitting a beta distribution (\(\alpha\) = number of events recorded by the array, \(\beta\) = number of events both known to have been missed and potentially missed by the array). and calculating the median estimated efficiency value using the R package actel [citation]."

3. If advEfficiency was run on an intra.array.CJS object:

   "Intra-array efficiency was estimated by comparing the tags detected at each of the two replicates. For each replicate, a beta distribution was fitted (\(\alpha\) = number of tags detected at both replicates, \(\beta\) = number of tags detected at the opposite replicate but not at the one for which efficiency is being calculated) and the median estimated efficiency value was calculated. The
overall efficiency of the array was then estimated as $1 - ((1 - R1) * (1 - R2))$, where $R1$ and $R2$ are the median efficiency estimates for each replicate. These calculations were performed using the R package actel [citation].

Replace [citation] with the output of `citation('actel')`

**Value**

A data frame with the required quantile values and a plot of the efficiency distributions.

**Examples**

```r
# Example using the output of simpleCJS.
x <- matrix(c(TRUE, TRUE, TRUE, TRUE, TRUE,
             TRUE, FALSE, TRUE, TRUE, FALSE,
             TRUE, TRUE, FALSE, FALSE, FALSE,
             TRUE, TRUE, FALSE, TRUE, TRUE,
             TRUE, TRUE, TRUE, FALSE, FALSE),
            ncol = 5, byrow = TRUE)
cjs.results <- simpleCJS(x)

# These cjs results can be used in advEfficiency
advEfficiency(cjs.results)

# Example using the output of dualArrayCJS.
x <- matrix(c(TRUE, TRUE,
             TRUE, FALSE,
             TRUE, TRUE,
             FALSE, TRUE),
            ncol = 2, byrow = TRUE)
colnames(x) <- c("A1.1", "A1.2")
cjs.results <- dualArrayCJS(x)

# These cjs results can be used in advEfficiency
advEfficiency(cjs.results)

# advEfficiency can also be run with the output from the main analyses.
# the example.results dataset is the output of a migration analysis
advEfficiency(example.results$overall.CJS)
```

**completeMatrix**  
*Complete a Distances Matrix*

**Description**

Completes the bottom diagonal of a matrix with the same number of rows and columns.
createWorkspace

Usage

completeMatrix(x)

Arguments

x A distances matrix to be completed.

Details

It is highly recommended to read the vignette regarding distances matrix before running this function. You can find it by running vignette('a-2_distances_matrix', 'actel') or browseVignettes('actel')

Value

A matrix of distances between pairs of points.

Examples

# Create dummy matrix with upper diagonal filled.
x <- matrix(
c( 0, 1, 2, 3, 4, 5,
   NA, 0, 1, 2, 3, 4,
   NA, NA, 0, 1, 2, 3,
   NA, NA, NA, 0, 1, 2,
   NA, NA, NA, NA, 0, 1,
   NA, NA, NA, NA, NA, 0),
ncol = 6, byrow = TRUE)

# inspect x
x

# run completeMatrix
completeMatrix(x)

createWorkspace Create a Default Workspace

Description

Produces template files and folders required to run the explore, migration and residency functions.

Usage

createWorkspace(dir)
Arguments

dir The name of the target directory. Will be created if not present.

Value

No return value, called for side effects

Examples

# running createWorkspace deploys template
# files to a directory specified by the user
createWorkspace(paste0(tempdir(), "/createWorkspace_example"))

Argument:

dataToList

Import RData in a list format

Description

Import RData in a list format

Usage

dataToList(source)

Arguments

source A RData file.

Value

A list containing the objects present in the source RData file.

Examples

# Dummy example:
# Create two objects:
object_1 <- "This"
object_2 <- "Worked!"

# Save them as an RData file in R’s temporary directory
save(object_1, object_2, file = paste0(tempdir(), "/dataToList_example.RData"))

# Remove the dummy objects as we don't need them any more
rm(object_1, object_2)

# Load the RData file as a single object
distancesMatrix

x <- dataToList(paste0(tempdir(), "/dataToList_example.RData"))

# inspect x
x

distancesMatrix  Calculate Distances Matrix

Description

Using a previously created transition layer (see transitionLayer), calculates the distances between spatial points. Adapted from Grant Adams' script "distance to closest mpa". If the argument 'actel' is set to TRUE (default), an actel-compatible matrix is generated, and the user will be asked if they would like to save the matrix as 'distances.csv' in the current directory.

Usage

distancesMatrix(
  t.layer,
  starters = NULL,
  targets = starters,
  coord.x = "x",
  coord.y = "y",
  id.col = NULL,
  actel = TRUE
)

Arguments

- **t.layer**: A TransitionLayer object, generated by transitionLayer.
- **starters**: A data frame with the points from which to start measuring the distance. Ignored if actel = TRUE (default), as the 'spatial.csv' is loaded as starters.
- **targets**: A data frame with the points to which a way must be found. Ignored if actel = TRUE (default), as the 'spatial.csv' is loaded as targets.
- **coord.x, coord.y**: The names of the columns containing the x and y coordinates in the starters and targets. Must be identical in the starters and targets.
- **id.col**: The name of the column containing the IDs of the points to be used as starters and targets. Must be identical in both files. Ignored if actel = TRUE (default), as the stations' standard names are used.
- **actel**: Logical: Should the distance matrix be optimized for actel? Defaults to TRUE.

Details

It is highly recommended to read the vignette regarding distances matrix before running this function. You can find it by running vignette('a-2_distances_matrix', 'actel') or browseVignettes('actel')
Value

A matrix with the distances between each pair of points.

Examples

```r
# check if R can run the distance functions
aux <- c(
  length(suppressWarnings(packageDescription("raster"))),
  length(suppressWarnings(packageDescription("gdistance"))),
  length(suppressWarnings(packageDescription("sp"))),
  length(suppressWarnings(packageDescription("tools"))),
  length(suppressWarnings(packageDescription("rgdal"))))
missing.packages <- sapply(aux, function(x) x == 1)
if (any(missing.packages)) {
  message("Sorry, this function requires packages ",
          paste(c("raster", "gdistance", "sp", "tools", "rgdal")[missing.packages], collapse = ", ",
                " to operate. Please install ", ifelse(sum(missing.packages) > 1, "them", "it"),
                " before proceeding.")
} else {
  if (suppressWarnings(require("rgdal"))) {
    # move to a temporary directory
    old.wd <- getwd()
    setwd(tempdir())

    # Fetch location of actel's example files
    aux <- system.file(package = "actel")[[1]]

    # create a temporary spatial.csv file
    file.copy(paste0(aux, "/example_spatial.csv"), "spatial.csv")

    # import the shape file and use the spatial.csv
    # to check the extents.
    x <- loadShape(path = aux, shape = "example_shapefile.shp",
                   coord.x = "x", coord.y = "y", size = 20)

    # Build the transition layer
    t.layer <- transitionLayer(x)

    # compile the distances matrix. Columns x and y in the spatial dataframe
    # contain the coordinates of the stations and release sites.
    distancesMatrix(t.layer, coord.x = 'x', coord.y = 'y')

    # return to original directory
    setwd(old.wd)
    rm(old.wd)
  } else {
    message("Sorry, it appears that rgdal is not being able to load.")
  }
}
rm(aux, missing.packages)
```
dualArrayCJS

Calculate estimated last-array efficiency

Description

Calculate estimated last-array efficiency

Usage

dualArrayCJS(input)

Arguments

input

A presence/absence matrix.

Value

A list containing:

- absolutes A matrix with the absolute number of tags detected at each replicate and at both,
- single.efficiency A vector of calculated array detection efficiencies for each of the replicates,
- combined.efficiency The value of the combined detection efficiency for the array.

References


Examples

# prepare a dummy presence/absence matrix
x <- matrix(c(TRUE, TRUE, TRUE, TRUE, FALSE, TRUE), ncol = 2)
colnames(x) <- c("R1", "R2")

# run CJS
dualArrayCJS(x)
emptyMatrix \hspace{1cm} \textit{Create a Template Distances Matrix} \\

\textbf{Description} \\
Creates an empty matrix based on the local 'spatial.csv' file and saves it to 'distances.csv' so the user can manually fill it.

\textbf{Usage} \\
\texttt{emptyMatrix(input = "spatial.csv")}

\textbf{Arguments} \\
\texttt{input} \hspace{1cm} Either a data frame with spatial data or the path to the file containing the spatial information.

\textbf{Details} \\
It is highly recommended to read the vignette regarding distances matrix before running this function. You can find it by running \texttt{vignette('a-2_distances_matrix', 'actel')} or \texttt{browseVignettes('actel')}.

\textbf{Value} \\
An empty matrix with the rows and columns required to operate with the target spatial file.

\textbf{Examples} \\
# This function requires a file with spatial information

# Fetch location of actel's example files
aux <- system.file(package = "actel")[1]

# run emptyMatrix on the temporary spatial.csv file
emptyMatrix(paste0(aux, "/example_spatial.csv"))

exampleWorkspace \hspace{1cm} \textit{Deploy Example Data} \\

\textbf{Description} \\
Creates a ready-to-run workspace with example data.

\textbf{Usage} \\
\texttt{exampleWorkspace(dir)}
explore

Arguments

dir The name of the target directory. Will be created if not present.

Value

No return value, called for side effects.

Examples

# deploy a minimal dataset to try actel!
exampleWorkspace(paste0(tempdir(), "/exampleWorkspace"))

---

explore Explorative Analysis

Description

explore allows you to quickly get a summary of your data. You can use explore to get a general feel for the study results, and check if the input files are behaving as expected. It is also a good candidate if you just want to validate your detections for later use in other analyses.

Usage

explore(
  tz = NULL,
  datapack = NULL,
  max.interval = 60,
  minimum.detections = 2,
  start.time = NULL,
  stop.time = NULL,
  speed.method = c("last to first", "last to last"),
  speed.warning = NULL,
  speed.error = NULL,
  jump.warning = 2,
  jump.error = 3,
  inactive.warning = NULL,
  inactive.error = NULL,
  exclude.tags = NULL,
  override = NULL,
  report = FALSE,
  auto.open = TRUE,
  discard.orphans = FALSE,
  discard.first = NULL,
  save.detections = FALSE,
GUI = c("needed", "always", "never"),
save.tables.locally = FALSE,
print.releases = TRUE,
plot.detections.by,
detections.y.axis = c("auto", "stations", "arrays")
)

Arguments

tz The time zone of the study area. Must match one of the values present in timezones.
datapack A data bundle pre-compiled through the function preload. May be used to run actel analyses based on R objects, rather than input files.
max.interval The number of minutes that must pass between detections for a new event to be created. Defaults to 60.
minimum.detections For tags with only one movement event, defines the minimum number of times a tag must have been recorded during the study period for it to be considered true detections and not random noise. Defaults to 2.
start.time Detection data prior to the timestamp set in start.time (in YYYY-MM-DD HH:MM:SS format) is not considered during the analysis.
stop.time Detection data posterior to the timestamp set in stop.time (in YYYY-MM-DD HH:MM:SS format) is not considered during the analysis.
speed.method Can take two forms: 'last to first' or 'last to last'. If 'last to first' (default), the last detection on the previous array is matched to the first detection on the target array to perform the calculations. If 'last to last', the last detection on the previous array is matched to the last detection on the target array to perform the calculations.
speed.warning If a tag moves at a speed equal or greater than speed.warning (in metres per second), a warning is issued. If left NULL (default), no warnings are issued.
speed.error If a tag moves at a speed equal or greater than speed.error (in metres per second), user intervention is suggested. If left NULL (default), user intervention is never suggested.
jump.warning If a tag crosses a number of arrays equal or greater than jump.warning without being detected, a warning is issued. If left NULL (default), no warnings are issued.
jump.error If a tag crosses a number of arrays equal or greater than jump.error without being detected, user intervention is suggested. If left NULL (default), user intervention is never suggested.
inactive.warning If a tag spends a number of days equal or greater than inactive.warning in a given array at the tail of the respective detections, a warning is issued. If left NULL (default), no warnings are issued.
inactive.error If a tag spends a number of days equal or greater than inactive.error in a given array at the tail of the respective detections, user intervention is suggested. If left NULL (default), user intervention is never suggested.
exclude.tags A vector of tags that should be excluded from the detection data before any analyses are performed. Intended to be used if stray tags from a different code space but with the same signal as a target tag are detected in the study area.

override A vector of signals for which the user intends to manually define which movement events are valid and invalid.

report Logical. Should an HTML report be created at the end of the analysis? NOTE: Setting report to TRUE will generate an HTML file in the current directory. Additionally, if auto.open = TRUE (default), the web browser will automatically be launched to open the report once the function terminates.

auto.open Logical: Should the report be automatically opened once the analysis is over? Defaults to TRUE. NOTE: If report = TRUE and auto.open = TRUE, the web browser will automatically be launched to open the report once the function terminates.

discard.orphans Logical: Should actel automatically discard detections that do not fall within receiver deployment periods, or that were recorded before the respective animals were released?

discard.first A threshold amount of time (in hours) that must pass after release for the respective detections to be valid. Set to 0 to discard only the release-to-first-detection calculations.

save.detections Logical: Should the processed detections be saved for future runs?

GUI One of "needed", "always" or "never". If "needed", a new window is opened to inspect the movements only when the movements table is too big to be displayed in R's console. If "always", a graphical interface is always created when the possibility to invalidate events emerges. If "never", a graphical interface is never invoked. In this case, if the table to be displayed does not fit in R's console, a temporary file will be saved and the user will be prompted to open that file and examine it. Defaults to "needed".

save.tables.locally Logical: If a table must be temporarily stored into a file for user inspection, should it be saved in the current working directory, or in R's temporary folder?

print.releases Logical: Should the release sites be printed in the study area diagrams?

plot.detections.by DEPRECATED. Please use the argument detections.y.axis instead.

detections.y.axis The type of y axis desired for the individual detection plots. While the argument defaults to "auto", it can be hard-set to one of "stations" or "arrays".

Value A list containing:

• bio: A copy of the biometrics input;
• detections: A list containing all detections for each target tag;
• valid.detections: A list containing the valid detections for each target tag;
• spatial: A list containing the spatial information used during the analysis;
• deployments: A data frame containing the deployments of each receiver;
• arrays: A list containing the array details used during the analysis;
• movements: A list containing all movement events for each target tag;
• valid.movements: A list containing the valid movement events for each target tag;
• times: A data frame containing all arrival times (per tag) at each array;
• rsp.info: A list containing containing appendix information for the RSP package;
• dist.mat: A matrix containing the distance matrix used in the analysis (if a valid distance matrix was supplied)

See Also
migration, residency

Examples

# Start by moving to a temporary directory
old.wd <- getwd()
setwd(tempdir())

# Deploy the example workspace
deployWorkspace("exampleWorkspace")

# Move your R session into the example workspace
setwd("exampleWorkspace")

# run the explore analysis. Ensure the tz argument
# matches the time zone of the study area. For the
# example dataset, tz = "Europe/Copenhagen"
results <- explore(tz = "Europe/Copenhagen")

# to obtain an HTML report, run the analysis with report = TRUE

# return to original working directory
setwd(old.wd)
rm(old.wd)
extractSignals

Usage

extractCodeSpaces(input)

Arguments

input A vector of transmitter names

Value

A vector of transmitter signals

Examples

# create dummy string
x <- c("R64K-1234", "A69-1303-12")

# run extractCodeSpaces
extractCodeSpaces(x)

extractSignals

Extract signals from transmitter names

Description

Extract signals from transmitter names

Usage

extractSignals(input)

Arguments

input A vector of transmitter names

Value

A vector of transmitter signals

Examples

# create dummy string
x <- c("R64K-1234", "A69-1303-12")

# run extractSignals
extractSignals(x)
getSpeeds

Extract speeds from the analysis results.

Description

Extract speeds from the analysis results.

Usage

getSpeeds(
  input,
  type = c("all", "forward", "backward"),
  direct = FALSE,
  n.events = c("first", "all", "last")
)

Arguments

input An actel results object generated by explore, migration or residency.
type The type of movements to record. One of "all", "forward", or "backward". In the two last options, only the forward or backwards (relatively to the study area structure) movement speeds are returned.
direct Logical: Extract only speeds between arrays that are directly connected (i.e. neighbouring arrays)?
n.events The events to record. One of "first", "all", or "last".

Value

A data frame with the following columns:

- Tag: The tag of the animal who performed the recorded speed
- Event: The valid event where the speed was recorded
- From.array: The array from which the tags left
- From.station: The station from which the tags left
- To.array: The array to which the tags arrived
- To.station: The station to which the tags arrived
- Speed: The speed recorded in the described movement

Examples

# using the example results loaded with actel
getSpeeds(example.results)

# You can specify which direction of movement to extract with 'type'
getSpeeds(example.results, type = "forward")


```r
# or
getspeeds(example.results, type = "backward")

# and also how many events per tag (this won’t change the output
# with the example.results, only because these results are minimal).
getspeeds(example.results, n.events = "first")
# or
getspeeds(example.results, n.events = "all")
# or
getspeeds(example.results, n.events = "last")
```

---

### Description

Extract timestamps from the analysis results.

### Usage

```r
getTimes(
  input, 
  locations = NULL, 
  move.type = c("array", "section"),
  event.type = c("arrival", "departure"),
  n.events = c("first", "all", "last")
)
```

### Arguments

- **input**
  An actel results object generated by `explore`, `migration` or `residency`.

- **locations**
  The names of the arrays or sections to be included. If left NULL, information for all arrays/sections is extracted.

- **move.type**
  The type of events to record: one of "array" or "section".

- **event.type**
  The point to be recorded: one of "arrival" or "departure".

- **n.events**
  The events to record. One of "first", "all", or "last".

### Value

A data frame with the timestamps for each tag (rows) and array (columns).
Examples

```r
# using the example results loaded with actel
getTimes(example.results)

# You can specify which events to extract with 'event.type'
getTimes(example.results, event.type = "arrival")
# or
getTimes(example.results, event.type = "departure")

# and also how many events per tag.
getTimes(example.results, n.events = "first")
# or
getTimes(example.results, n.events = "all")
# or
getTimes(example.results, n.events = "last")
```

loadShape

Load shapefile and convert to a raster object.

Description

loadShape can also perform early quality checks on the shape file, to ensure it is compatible with the remaining study data. To activate these, set the names of the columns in the spatial.csv file that contain the x and y coordinates of the stations using coord.x and coord.y. By default, loadShape looks for a spatial.csv file in the current working directory, but this can be personalized using the spatial argument.

Usage

```r
loadShape(
  path = ".",
  shape,
  size,
  spatial = "spatial.csv",
  coord.x = NULL,
  coord.y = NULL,
  buffer = NULL,
  type = c("land", "water")
)
```

Arguments

- **path** (The system path to the 'shape' file. Defaults to the current directory.)
- **shape** (A shape file containing land polygons of the study area.)
- **size** (The pixel size, in metres.)
loadShape

Either a character string specifying the path to a spatial.csv file or a spatial data frame. This argument is not mandatory, but can be used to perform an early check of the shape file’s compatibility with the study stations and release sites.

coord.x, coord.y
The names of the columns containing the x and y positions of the stations in the spatial.csv file. These coordinates MUST BE in the same coordinate system as the shape file.

buffer
Artificially expand the shape file edges. Can be a single value (applied to all edges) or four values (xmin, xmax, ymin, ymax). The unit of the buffer depends on the shape file’s coordinate system.

type
The type of shapefile being loaded. One of "land" or "water".

Details
It is highly recommended to read the vignette regarding distances matrix before running this function. You can find it by running vignette("a-2_distances_matrix", 'actel') or browseVignettes('actel')

Value
A raster object.

Examples

# check if R can run the distance functions
aux <- c(
  length(suppressWarnings(packageDescription("raster"))),
  length(suppressWarnings(packageDescription("gdistance"))),
  length(suppressWarnings(packageDescription("sp"))),
  length(suppressWarnings(packageDescription("tools"))),
  length(suppressWarnings(packageDescription("rgdal"))))
missing.packages <- sapply(aux, function(x) x == 1)
if (any(missing.packages)) {
  message("Sorry, this function requires packages ",
          paste(c("raster", "gdistance", "sp", "tools", "rgdal")[missing.packages], collapse = "", "")
          " to operate. Please install ", ifelse(sum(missing.packages) > 1, "them", "it"),
          " before proceeding."
) } else {
  if (suppressWarnings(require("rgdal"))) {
    # Fetch actel’s example shapefile location
    aux <- system.file(package = "actel")[1]
    # import the shape file
    x <- loadShape(path = aux, shape = "example_shapefile.shp", size = 20)
    # have a look at the resulting raster,
    # where the blank spaces are the land areas
    raster::plot(x)
  } else {
    message("Sorry, it appears that rgdal is not being able to load."
)
Description

Loads a spatial file prepared for actel and appends the Standard.name column. Additionally, performs a series of quality checks on the contents of the target file.

Usage

loadSpatial(input = "spatial.csv", section.order = NULL)

Arguments

input Either a data frame or the name of an input file with spatial data in the actel format.
section.order A vector containing the order by which sections should be aligned in the results.

Value

A data frame with the spatial information present in 'spatial.csv' and the Standard.name column.

Examples

# This function requires the presence of a file with spatial information

# Fetch location of actel's example files
aux <- system.file(package = "actel")[1]

# run loadSpatial on the temporary spatial.csv file
loadSpatial(input = paste0(aux, '/example.spatial.csv'))
Description

The migration analysis runs the same initial checks as explore, but on top of it, it analyses the animal behaviour. By selecting the arrays that lead to success, you can define whether or not your animals survived the migration. Additional plots help you find out if some animal/tag has been acting odd. Multiple options allow you to tweak the analysis to fit your study perfectly.

Usage

migration(
  tz = NULL,
  section.order = NULL,
  datapack = NULL,
  success.arrays = NULL,
  max.interval = 60,
  minimum.detections = 2,
  start.time = NULL,
  stop.time = NULL,
  speed.method = c("last to first", "last to last"),
  speed.warning = NULL,
  speed.error = NULL,
  jump.warning = 2,
  jump.error = 3,
  inactive.warning = NULL,
  inactive.error = NULL,
  exclude.tags = NULL,
  override = NULL,
  report = FALSE,
  auto.open = TRUE,
  discard.orphans = FALSE,
  discard.first = NULL,
  save.detections = FALSE,
  if.last.skip.section = TRUE,
  replicates = NULL,
  disregardparallels = TRUE,
  GUI = c("needed", "always", "never"),
  save.tables.locally = FALSE,
  print.releases = TRUE,
  plot.detections.by,
  detections.y.axis = c("auto", "stations", "arrays")
)
Arguments

tz The time zone of the study area. Must match one of the values present in `timezones`.

section.order A vector containing the order by which sections should be aligned in the results.

datapack A data bundle pre-compiled through the function `preload`. May be used to run actel analyses based on R objects, rather than input files.

success.arrays The arrays that mark the end of the study area. If a tag crosses one of these arrays, the respective animal is considered to have successfully migrated through the study area.

max.interval The number of minutes that must pass between detections for a new event to be created. Defaults to 60.

minimum.detections For tags with only one movement event, defines the minimum number of times a tag must have been recorded during the study period for it to be considered true detections and not random noise. Defaults to 2.

start.time Detection data prior to the timestamp set in `start.time` (in YYYY-MM-DD HH:MM:SS format) is not considered during the analysis.

stop.time Detection data posterior to the timestamp set in `stop.time` (in YYYY-MM-DD HH:MM:SS format) is not considered during the analysis.

speed.method Can take two forms: 'last to first' or 'last to last'. If 'last to first' (default), the last detection on the previous array is matched to the first detection on the target array to perform the calculations. If 'last to last', the last detection on the previous array is matched to the last detection on the target array to perform the calculations.

speed.warning If a tag moves at a speed equal or greater than `speed.warning` (in metres per second), a warning is issued. If left NULL (default), no warnings are issued.

speed.error If a tag moves at a speed equal or greater than `speed.error` (in metres per second), user intervention is suggested. If left NULL (default), user intervention is never suggested.

jump.warning If a tag crosses a number of arrays equal or greater than `jump.warning` without being detected, a warning is issued. If left NULL (default), no warnings are issued.

jump.error If a tag crosses a number of arrays equal or greater than `jump.error` without being detected, user intervention is suggested. If left NULL (default), user intervention is never suggested.

inactive.warning If a tag spends a number of days equal or greater than `inactive.warning` in a given array at the tail of the respective detections, a warning is issued. If left NULL (default), no warnings are issued.

inactive.error If a tag spends a number of days equal or greater than `inactive.error` in a given array at the tail of the respective detections, user intervention is suggested. If left NULL (default), user intervention is never suggested.

exclude.tags A vector of tags that should be excluded from the detection data before any analyses are performed. Intended to be used if stray tags from a different code space but with the same signal as a target tag are detected in the study area.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>override</td>
<td>A vector of signals for which the user intends to manually define which movement events are valid and invalid.</td>
</tr>
<tr>
<td>report</td>
<td>Logical. Should an HTML report be created at the end of the analysis? NOTE: Setting report to TRUE will generate an HTML file in the current directory. Additionally, if auto.open = TRUE (default), the web browser will automatically be launched to open the report once the function terminates.</td>
</tr>
<tr>
<td>auto.open</td>
<td>Logical: Should the report be automatically opened once the analysis is over? Defaults to TRUE. NOTE: If report = TRUE and auto.open = TRUE, the web browser will automatically be launched to open the report once the function terminates.</td>
</tr>
<tr>
<td>discard.orphans</td>
<td>Logical: Should Actel automatically discard detections that do not fall within receiver deployment periods, or that were recorded before the respective animals were released?</td>
</tr>
<tr>
<td>discard.first</td>
<td>A threshold amount of time (in hours) that must pass after release for the respective detections to be valid. Set to 0 to discard only the release-to-first-detection calculations.</td>
</tr>
<tr>
<td>save.detections</td>
<td>Logical: Should the processed detections be saved for future runs?</td>
</tr>
<tr>
<td>if.last.skip.section</td>
<td>Logical: Should a tag detected at the last array of a given section be considered to have disappeared in the next section?</td>
</tr>
<tr>
<td>replicates</td>
<td>A list containing, for each array to which intra-array efficiency is to be calculated: The standard names of the stations to be used as a replicate. See the vignettes for more details.</td>
</tr>
<tr>
<td>disregard.parallels</td>
<td>Logical: Should the presence of parallel arrays invalidate potential efficiency peers? See the vignettes for more details.</td>
</tr>
<tr>
<td>GUI</td>
<td>One of &quot;needed&quot;, &quot;always&quot; or &quot;never&quot;. If &quot;needed&quot;, a new window is opened to inspect the movements only when the movements table is too big to be displayed in R’s console. If &quot;always&quot;, a graphical interface is always created when the possibility to invalidate events emerges. If &quot;never&quot;, a graphical interface is never invoked. In this case, if the table to be displayed does not fit in R’s console, a temporary file will be saved and the user will be prompted to open that file and examine it. Defaults to &quot;needed&quot;.</td>
</tr>
<tr>
<td>save.tables.locally</td>
<td>Logical: If a table must be temporarily stored into a file for user inspection, should it be saved in the current working directory, or in R’s temporary folder?</td>
</tr>
<tr>
<td>print.releases</td>
<td>Logical: Should the release sites be printed in the study area diagrams?</td>
</tr>
<tr>
<td>plot.detections.by</td>
<td>DEPRECATED. Please use the argument detections.y.axis instead.</td>
</tr>
<tr>
<td>detections.y.axis</td>
<td>The type of y axis desired for the individual detection plots. While the argument defaults to &quot;auto&quot;, it can be hard-set to one of &quot;stations&quot; or &quot;arrays&quot;.</td>
</tr>
</tbody>
</table>
Value

A list containing:

- detections: A list containing all detections for each target tag;
- valid.detections: A list containing the valid detections for each target tag;
- spatial: A list containing the spatial information used during the analysis;
- deployments: A data frame containing the deployments of each receiver;
- arrays: A list containing the array details used during the analysis;
- movements: A list containing all movement events for each target tag;
- valid.movements: A list containing the valid movement events for each target tag;
- section.movements: A list containing the valid section shifts for each target tag;
- status.df: A data.frame containing summary information for each tag, including the following columns:
  - Times.entered.[section]: Number of times the tag was recorded entering a given section.
  - Average.time.until.[section]: Time spent between release or leaving another section and reaching at the given section.
  - Average.speed.to.[section]: Average speed from release or leaving one section and reaching the given section (if speed.method = "last to first"), or from release/leaving one section and leaving the given section (if speed.method = "last to last").
  - First.array.[section]: Array in which the tag was first detected in a given section
  - First.station.[section]: Standard name of the first station where the tag was detected in a given section
  - First.arrived.[section]: Very first arrival time at a given section
  - Average.time.in.[section]: Average time spent within a given section at each stay.
  - Average.speed.in.[section]: Average speed within a given section at each stay (only displayed if speed.method = "last to first").
  - Last.array.[section]: Array in which the tag was last detected in a given section
  - Last.station.[section]: Standard name of the last station where the tag was detected in a given section
  - Last.left.[section]: Very last departure time from a given section
  - Total.time.in.[section]: Total time spent in a given section
  - Very.last.array: Last array where the tag was detected
  - Status: Fate assigned to the tag
  - Valid.detections: Number of valid detections
  - Invalid.detections: Number of invalid detections
  - Backwards.movements: Number of backward movement events
  - Max.cons.back.moves: Longest successive backwards movements
  - P.type: Type of processing:
    * 'Skipped' if no data was found for the tag,
    * 'Auto' if no user interaction was required,
    * 'Manual' if user interaction was suggested and the user made changes to the validity of the events,
"Overridden" if the user listed the tag in the \texttt{override} argument.

- **Comments**: Comments left by the user during the analysis

- **section\_overview**: A data frame containing the number of tags that disappeared in each section;
- **group\_overview**: A list containing the number of known and estimated tags to have passed through each array, divided by group;
- **release\_overview**: A list containing the number of known and estimated tags to have passed through each array, divided by group and release sites;
- **matrices**: A list of CJS matrices used for the efficiency calculations;
- **overall\_CJS**: A list of CJS results of the inter-array CJS calculations;
- **intra\_array\_CJS**: A list of CJS results of the intra-array CJS calculations;
- **times**: A data frame containing all arrival times (per tag) at each array;
- **rsp\_info**: A list containing appendix information for the RSP package;
- **dist\_mat**: The distance matrix used in the analysis (if a valid distance matrix was supplied)

**See Also**

\texttt{explore}, \texttt{residency}

**Examples**

```r
# Start by moving to a temporary directory
old.wd <- getwd()
setwd(tempdir())

# Deploy the example workspace
eexampleWorkspace("exampleWorkspace")

# Move your R session into the example workspace
setwd("exampleWorkspace")

# run the migration analysis. Ensure the \texttt{tz} argument
# matches the time zone of the study area and that the
# sections match your array names. The line below works
# for the example data.
results <- migration(tz = "Europe/Copenhagen")

# to obtain an HTML report, run the analysis with \texttt{report} = \texttt{TRUE}

# return to original working directory
setwd(old.wd)
rm(old.wd)
```
plotArray

Plot simultaneous/cumulative presences at a give array or set of arrays

Description

Plot simultaneous/cumulative presences at a give array or set of arrays

Usage

plotArray(
  input,
  arrays,
  title,
  xlab, ylab,
  lwd = 1,
  col = "#56B4E9",
  type = c("default", "bars", "lines"),
  timestep = c("days", "hours", "mins"),
  cumulative = FALSE
)

Arguments

input The results of an actel analysis (either explore, migration or residency).
arrays One or more arrays to be analysed. If multiple arrays are provided, data will be grouped.
title An optional title for the plot.
xlab, ylab Optional axis names for the plot. If left empty, default axis names will be added.
lwd The line width, only relevant for line plots.
col The colour of the line or bars. Defaults to blue.
type The type of plot to be drawn. By default, a line is plotted if cumulative = TRUE, and bars are plotted otherwise.
timestep The time resolution for the grouping of the results. Defaults to "days", but can be set to "hours" and "mins" (at the expense of computing time).
cumulative Logical. If TRUE, a cumulative plot of arrivals is drawn, otherwise the number of tags simultaneously present at the array(s) is drawn.

Value

A ggplot object.
Examples

# Using the example results that come with actel
plotArray(example.results, arrays = "A9")

# Because plotArray returns a ggplot object, you can store
# it and edit it manually, e.g.:  
library(ggplot2)
P <- plotArray(example.results, arrays = "A9")
P <- P + xlab("changed the x axis label a posteriori")
P
# You can also save the plot using ggsave!

plotDetections

Plot detections for a single tag

Description

The output of plotDetections is a ggplot object, which means you can then use it in combination
with other ggplot functions, or even together with other packages such as patchwork.

Usage

plotDetections(
  input,
  tag,
  type,
  y.axis = c("auto", "stations", "arrays"),
  title,
  xlab,
  ylab,
  col,
  array.alias,
  section.alias,
  frame.warning = TRUE,
  x.label.format,
  only.valid = FALSE,
  like.migration = TRUE
)

Arguments

input The results of an actel analysis (either explore, migration or residency).
tag The transmitter to be plotted.
type DEPRECATED. Please use the argument y.axis instead.
y.axis The type of y axis desired. One of "stations" (default) or "arrays".
plotLive

Plot array live times

**Description**

Plot array live times

<table>
<thead>
<tr>
<th>title</th>
<th>An optional title for the plot. If left empty, a default title will be added.</th>
</tr>
</thead>
<tbody>
<tr>
<td>xlab, ylab</td>
<td>Optional axis names for the plot. If left empty, default axis names will be added.</td>
</tr>
<tr>
<td>col</td>
<td>An optional colour scheme for the detections. If left empty, default colours will be added.</td>
</tr>
<tr>
<td>array.alias</td>
<td>A named vector of format c(&quot;old_array_name&quot; = &quot;new_array_name&quot;) to replace default array names with user defined ones.</td>
</tr>
<tr>
<td>section.alias</td>
<td>A named vector of format c(&quot;old_section_name&quot; = &quot;new_section_name&quot;) to replace default section names with user defined ones.</td>
</tr>
<tr>
<td>frame.warning</td>
<td>Logical. By default, actel highlights manually changed or overridden tags in yellow and red plot frames, respectively. Set to FALSE to deactivate this behaviour.</td>
</tr>
<tr>
<td>x.label.format</td>
<td>A character string giving a date-time format for the x labels. If missing, ggplot’s default labels are used.</td>
</tr>
<tr>
<td>only.valid</td>
<td>Logical. Should only valid detections be printed?</td>
</tr>
<tr>
<td>like.migration</td>
<td>Logical. For plots originating from migration analyses, should the additional grey vertical bars be included? Defaults to TRUE, and only has a visible effect if the input stems from a migration analysis.</td>
</tr>
</tbody>
</table>

**Value**

A ggplot object.

**Examples**

```r
# Using the example results that come with actel
plotDetections(example.results, 'R64K-4451')

# Because plotDetections returns a ggplot object, you can store
# it and edit it manually, e.g.:
library(ggplot2)
p <- plotDetections(example.results, 'R64K-4451')
p <- p + xlab("changed the x axis label a posteriori")
p

# You can also save the plot using ggsave!
```
plotLive

Usage

plotLive(
  input,  # An actel results object, or a preload object
  arrays,  # Optional: A subset of arrays to be plotted
  show.stations = FALSE,  # Logical: Should the live times of each station be shown under the array bars?
  array.size = 2,  # The size of the array bars (defaults to 2)
  station.size = 1,  # The size of the station bars (defaults to 1)
  show.caps = TRUE,  # Logical: Should cap lines be shown at the end of each live period?
  cap.prop = 2,  # The relative size of the caps, as compared to the respective bars (defaults to 2).
  title = "",  # An optional title for the plot.
  xlab = "",  # Optional axis names for the plot.
  ylab = "",  # An optional colour scheme for the array bars. If left empty, default colours will
  col  # be added. Note: Station bars are 40% lighter than the array bars.
)

Arguments

input  An actel results object, or a preload object
arrays Optional: A subset of arrays to be plotted
show.stations Logical: Should the live times of each station be shown under the array bars?
array.size The size of the array bars (defaults to 2)
station.size The size of the station bars (defaults to 1)
show.caps Logical: Should cap lines be shown at the end of each live period?
cap.prop The relative size of the caps, as compared to the respective bars (defaults to 2).
title An optional title for the plot.
xlab, ylab Optional axis names for the plot.
col An optional colour scheme for the array bars. If left empty, default colours will be added. Note: Station bars are 40% lighter than the array bars.

Value

A ggplot object.

Examples

# Using the example results that come with actel
plotLive(example.results)

# Because plotLive returns a ggplot object, you can store it and edit it manually, e.g.:  
library(ggplot2)
p <- plotLive(example.results)
p <- p + xlab("changed the x axis label a posteriori")
p
# You can also save the plot using ggsave!
plotMoves

Plot moves for one ore more tags

Description

The output of plotMoves is a ggplot object, which means you can then use it in combination with other ggplot functions, or even together with other packages such as patchwork.

Usage

plotMoves(
  input,
  tags, 
  title, 
  xlab, 
  ylab, 
  col,
  array.alias, 
  show.release = TRUE
)

Arguments

input The results of an actel analysis (either explore, migration or residency).
tags The transmitters to be plotted (optional).
title An optional title for the plot.
xlab, ylab Optional axis names for the plot. If left empty, default axis names will be added.
col An optional colour scheme for the detections. If left empty, default colours will be added.
array.alias A named vector of format c("old_array_name" = "new_array_name") to replace default array names with user defined ones.
show.release Logical: Should the line from release to first detection be displayed?

Value

A ggplot object.

Examples

# Using the example results that come with actel
plotMoves(example.results, 'R64K-4451')

# Because plotMoves returns a ggplot object, you can store
# it and edit it manually, e.g.:
library(ggplot2)
p <- plotMoves(example.results, 'R64K-4451')
plotRatios

p <- p + xlab("changed the x axis label a posteriori")
p

# You can also save the plot using ggsave!

---

**Description**

By default, this function plots the global residency. However, you can use the argument 'group' to plot the results only from a specific animal group. Lastly, you can also use 'sections', rather than 'group', to compare the residency at a specific section (or group of sections) between the different groups.

**Usage**

```r
plotRatios(
  input,
  group,
  sections,
  type = c("absolutes", "percentages"),
  title,
  xlab,
  ylab,
  col
)
```

**Arguments**

- **input**: The results of an actel analysis (either explore, migration or residency).
- **group**: An optional argument to plot only the data corresponding to one group.
- **sections**: An optional argument to plot the residency of the multiple groups for a specific subset of sections.
- **type**: The type of residency to be displayed. One of 'absolutes' (the default) or 'percentages'.
- **title**: An optional title for the plot. If left empty, a default title will be added.
- **xlab, ylab**: Optional axis names for the plot. If left empty, default axis names will be added.
- **col**: An optional colour scheme for the detections. If left empty, default colours will be added.

**Details**

The output of plotRatios is a ggplot object, which means you can then use it in combination with other ggplot functions, or even together with other packages such as patchwork.
plotResidency

Plot residency for a single tag

Description

The output of plotResidency is a ggplot object, which means you can then use it in combination with other ggplot functions, or even together with other packages such as patchwork.

Usage

plotResidency(input, tag, title, xlab, ylab, col)

Arguments

input The results of an actel analysis (either explore, migration or residency).
tag The transmitter to be plotted.
title An optional title for the plot. If left empty, a default title will be added.
xlab, ylab Optional axis names for the plot. If left empty, default axis names will be added.
col An optional colour scheme for the detections. If left empty, default colours will be added.

Value

A ggplot object.
Examples

# For this example, I have modified the example.results that come with actel,
# so they resemble a residency output

plotResidency(example.residency.results, 'R64K-4451')

# Because plotResidency returns a ggplot object, you can store
# it and edit it manually, e.g.:
library(ggplot2)
p <- plotResidency(example.residency.results, 'R64K-4451')
p <- p + xlab("changed the x axis label a posteriori")
p

# You can also save the plot using ggsave!

plotSensors

Plot sensor data for a single tag

Description

The output of plotSensors is a ggplot object, which means you can then use it in combination with other ggplot functions, or even together with other packages such as patchwork.

Usage

plotSensors(
  input, tag, sensor, title = tag, xlab, ylab, pcol, psize = 1, lsize = 0.5, colour.by = c("array", "section"), array.alias, lcol = "grey40", verbose = TRUE
)

Arguments

input The results of an actel analysis (either explore, migration or residency).
tag The transmitter to be plotted.
sensor The sensors to be plotted. If left empty, all available sensors are plotted.
**plotTimes**

Print circular graphics for time series.

**Description**

Wraps functions adapted from the circular R package.

**Usage**

```r
plotTimes(
  times,
  night = NULL,
  col,
  alpha = 0.8,
  title = "",
)
```

**Value**

A ggplot object.

**Examples**

```r
# Using the example results that come with actel
plotSensors(example.results, 'R64K-4451')

# Because plotSensors returns a ggplot object, you can store
# it and edit it manually, e.g.:
library(ggplot2)
p <- plotSensors(example.results, 'R64K-4451')
p <- p + xlab("changed the x axis label a posteriori")
p

# You can also save the plot using ggsave!
```
mean.dash = TRUE,
mean.range = TRUE,
mean.range.darken.factor = 1.4,
rings = TRUE,
file,
width,
height,
bg = "transparent",
col,
legend.pos = c("auto", "corner", "bottom"),
ylegend,
xlegend,
xjust = c("auto", "centre", "left", "right"),
exand = 0.95,
cex = 1
)

Arguments

times            A list of of time vectors (each vector will be plotted as a series).
night            A vector of two times defining the start and stop of the night period (in HH:MM format).
col              A vector of colour names to paint each time series (colours will be added transparency).
alpha            A value between 0 and 1 for the opacity of each layer (defaults to 0.8).
title            A title for the plot.
mean.dash        Logical: Should the mean value be displayed on the plot’s edge?
mean.range       Logical: Should the SEM be displayed? (only relevant if mean.dash = TRUE)
mean.range.darken.factor
                   A numeric factor to darken the mean range edges for each group. Values greater than 1 darken the colour, and values lower than 1 lighten the colour.
rings            Logical: Should inner plot rings be displayed?
file             A file name to save the plot to. Leave NULL to plot on active graphics device. Available file extensions: .svg, .pdf, .png and .tiff.
height, width    The height and width of the output file. Use inches for .pdf and .svg files or pixels for .png and .tiff files.
bg               The colour of the plot background. Defaults to "transparent".
col              The number of columns in which to set the legend items. By default, actel decides the number of columns based on the number of data series to be plotted.
legend.pos       Where should the legend be drawn? By default, actel decides whether to plot the legend in the corner of the plot at the bottom the plot depending on the number of data series to plot. Possible values: 'auto', 'corner', 'bottom'.
ylegend          Adjustment to the vertical positioning of the legend. Only relevant if the legend is being drawn in the corner of the plot.
xlegend
  Adjustment to the horizontal positioning of the legend.

xjust
  How the legend is to be justified when the legend is drawn at the bottom of the plot. One of 'auto' (i.e. let actel decide the best adjustment), 'left', 'centre', or 'right'.

expand
  Parameter that controls the size of the plotted circle. Defaults to 0.95. Larger values expand the circle, while smaller values shrink the circle.

cex
  A numerical vector giving the amount by which plotting characters and symbols should be scaled relative to the default. When saving the plot in a vectorial form, it is recommended to change the height and width arguments rather than the cex.

Details

For more details about the original functions, visit the circular package homepage at https://github.com/cran/circular

Value

A circular plot

Examples

# The output of timesToCircular can be used as an input to plotTimes.
x <- getTimes(example.results, location = "A1", n.events = "first", event.type = "arrival")
times <- timesToCircular(x)

# plot times
plotTimes(times)

# A night period can be added with 'night'
plotTimes(times, night = c("20:00", "06:00"))

Description

This function allows the user to prepare a set of R objects to be run through an explore, migration or residency analysis.

Usage

preload(
  biometrics,
  spatial,
  deployments,
  detections,
dot,
distances,
tz,
start.time = NULL,
stop.time = NULL,
section.order = NULL,
exclude.tags = NULL,
disregard.parallels = FALSE,
discard.orphans = FALSE
)

Arguments

biometrics A data frame containing biometric information.
spatial A data frame containing spatial information.
deployments A data frame containing deployment information.
detections A data frame containing the detections.
dot A DOT string of the array configuration.
distances A distances matrix between arrays. See distancesMatrix.
tz The time zone of the study area. Must match one of the values present in timezones.
start.time Detection data prior to the timestamp set in start.time (in YYYY-MM-DD HH:MM:SS format) is not considered during the analysis.
stop.time Detection data posterior to the timestamp set in stop.time (in YYYY-MM-DD HH:MM:SS format) is not considered during the analysis.
section.order A vector containing the order by which sections should be aligned in the results.
exclude.tags A vector of tags that should be excluded from the detection data before any analyses are performed. Intended to be used if stray tags from a different code space but with the same signal as a target tag are detected in the study area.
disregard.parallels Logical: Should the presence of parallel arrays invalidate potential efficiency peers? See the vignettes for more details.
discard.orphans Logical: Should actel automatically discard detections that do not fall within receiver deployment periods, or that were recorded before the respective animals were released?

Value

A dataset that can be used as an input for actel’s main analyses. This dataset contains:

- **bio**: The biometric data
- **sections**: The sections of the study area, if set using the argument sections (required to run residency and migration analyses)
- **deployments**: The deployment data
• spatial: The spatial data, split in stations and release sites.
• dot: A table of array connections.
• arrays: A list with the details of each array
• dotmat: A matrix of distances between arrays (in number of arrays).
• dist.mat: The distances matrix.
• detections.list: A processed list of detections for each tag.
• paths: A list of the possible paths between each pair of arrays.
• disregard.parallels: Logical: Should parallel arrays invalidate efficiency peers? (required to run residency and migration analyses)
• tz: The time zone of the study area

Examples

# This function requires a series of pre-created R objects.
# We can create them by loading the example files from actel:
aux <- system.file(package = "actel")[1]
bio <- read.csv(paste0(aux, "/example_biometrics.csv"))
deployments <- read.csv(paste0(aux, "/example_deployments.csv"))
spatial <- read.csv(paste0(aux, "/example_spatial.csv"))
detections <- read.csv(paste0(aux, "/example_detections.csv"))
dot <- "A0--A1--A2--A3--A4--A5--A6--A7--A8--A9"

# Now that we have the R objects created, we can run preload:
x <- preload(biometrics = bio, deployments = deployments, spatial = spatial,
             detections = detections, dot = dot, tz = "Europe/Copenhagen")

---

readDot

Read dot file or string

Description

Read dot file or string

Usage

readDot(input = NULL, string = NULL, silent = FALSE)

Arguments

input The name of a file containing dot connections.
string A string of dot connections.
silent Logical: Should warnings be suppressed?
Value

A data frame with the connections present in the input.

Examples

```r
# create dummy dot string
x1 <- c("A--B--C--D--E--F")

# run readDot
readDot(string = x1)

# more complex strings are acceptable:
x2 <- c("A--B--C--D--E--F
A--G--H--I--E
H--C")

readDot(string = x2)

# Alternatively, connections can be read from a file

# let's create a dummy file in R's temporary directory:
write("A--B--C--D--E--F\nA--G--H--I--E\nH--C",
     file = paste0(tempdir(), "/dummy_dot.txt"))

# now we can read it using readDot
readDot(input = paste0(tempdir(), "/dummy_dot.txt"))
```

recoverLog

Recover latest actel crash log

Description

Recover latest actel crash log

Usage

```r
recoverLog(file, overwrite = FALSE)
```

Arguments

- **file**: Name of the file to which the log should be saved.
- **overwrite**: Logical: If 'file' already exists, should its content be overwritten?

Value

No return value, called for side effects.
Examples

recoverLog(file = paste0(tempdir(), "/new_log.txt"))

---

**residency**

**Residency Analysis**

Description

The **residency** analysis runs the same initial checks as **explore**, but, similarly to **migration**, explores particular points of the animal behaviour. If you want to know where your animals were in each day of the study, how many animals were in each section each day, and other residency-focused variables, this is the analysis you are looking for!

Usage

```r
residency(
  tz = NULL,
  section.order = NULL,
  datapack = NULL,
  max.interval = 60,
  minimum.detections = 2,
  start.time = NULL,
  stop.time = NULL,
  speed.method = c("last to first", "last to last"),
  speed.warning = NULL,
  speed.error = NULL,
  jump.warning = 2,
  jump.error = 3,
  inactive.warning = NULL,
  inactive.error = NULL,
  exclude.tags = NULL,
  override = NULL,
  report = FALSE,
  auto.open = TRUE,
  discard.orphans = FALSE,
  discard.first = NULL,
  save.detections = FALSE,
  section.minimum = 2,
  timestep = c("days", "hours"),
  replicates = NULL,
  GUI = c("needed", "always", "never"),
  save.tables.locally = FALSE,
)```
print.releases = TRUE,
plot.detections.by,
detections.y.axis = c("auto", "stations", "arrays")
)

Arguments

tz The time zone of the study area. Must match one of the values present in
timezones.

section.order A vector containing the order by which sections should be aligned in the results.

datapack A data bundle pre-compiled through the function preload. May be used to run
actel analyses based on R objects, rather than input files.

max.interval The number of minutes that must pass between detections for a new event to be
created. Defaults to 60.

minimum.detections For tags with only one movement event, defines the minimum number of times
a tag must have been recorded during the study period for it to be considered
ture detections and not random noise. Defaults to 2.

start.time Detection data prior to the timestamp set in start.time (in YYYYY-MM-DD
HH:MM:SS format) is not considered during the analysis.

stop.time Detection data posterior to the timestamp set in stop.time (in YYYYY-MM-DD
HH:MM:SS format) is not considered during the analysis.

speed.method Can take two forms: 'last to first' or 'last to last'. If 'last to first' (default),
the last detection on the previous array is matched to the first detection on the
target array to perform the calculations. If 'last to last', the last detection on 'the
previous array is matched to the last detection on the target array to perform the
calculations.

speed.warning If a tag moves at a speed equal or greater than speed.warning (in metres per
second), a warning is issued. If left NULL (default), no warnings are issued.

speed.error If a tag moves at a speed equal or greater than speed.error (in metres per
second), user intervention is suggested. If left NULL (default), user intervention
is never suggested.

jump.warning If a tag crosses a number of arrays equal or greater than jump.warning without
being detected, a warning is issued. If left NULL (default), no warnings are
issued.

jump.error If a tag crosses a number of arrays equal or greater than jump.error without
being detected, user intervention is suggested. If left NULL (default), user in-
tervention is never suggested.

inactive.warning If a tag spends a number of days equal or greater than inactive.warning in a
given array at the tail of the respective detections, a warning is issued. If left
NULL (default), no warnings are issued.

inactive.error If a tag spends a number of days equal or greater than inactive.error in a
given array at the tail of the respective detections, user intervention is suggested.
If left NULL (default), user intervention is never suggested.
exclude.tags A vector of tags that should be excluded from the detection data before any analyses are performed. Intended to be used if stray tags from a different code space but with the same signal as a target tag are detected in the study area.

event.areas A vector of signals for which the user intends to manually define which movement events are valid and invalid.

report Logical. Should an HTML report be created at the end of the analysis? NOTE: Setting report to TRUE will generate an HTML file in the current directory. Additionally, if auto.open = TRUE (default), the web browser will automatically be launched to open the report once the function terminates.

auto.open Logical: Should the report be automatically opened once the analysis is over? Defaults to TRUE. NOTE: If report = TRUE and auto.open = TRUE, the web browser will automatically be launched to open the report once the function terminates.

discard.orphans Logical: Should actel automatically discard detections that do not fall within receiver deployment periods, or that were recorded before the respective animals were released?

discard.first A threshold amount of time (in hours) that must pass after release for the respective detections to be valid. Set to 0 to discard only the release-to-first-detection calculations.

save.detections Logical: Should the processed detections be saved for future runs?

section.minimum If a tag has less than section.minimum consecutive detections in a section, a warning is issued. Defaults to 2.

timestep The resolution desired for the residency calculations. One of "days" (default) or "hours".

replicates A list containing, for each array to which intra-array efficiency is to be calculated: The standard names of the stations to be used as a replicate. See the vignettes for more details.

GUI One of "needed", "always" or "never". If "needed", a new window is opened to inspect the movements only when the movements table is too big to be displayed in R’s console. If "always", a graphical interface is always created when the possibility to invalidate events emerges. If "never", a graphical interface is never invoked. In this case, if the table to be displayed does not fit in R’s console, a temporary file will be saved and the user will be prompted to open that file and examine it. Defaults to "needed".

save.tables.locally Logical: If a table must be temporarily stored into a file for user inspection, should it be saved in the current working directory, or in R’s temporary folder?

print.releases Logical: Should the release sites be printed in the study area diagrams?

plot.detections.by DEPRECATED. Please use the argument detections.y.axis instead.

detections.y.axis The type of y axis desired for the individual detection plots. While the argument defaults to "auto", it can be hard-set to one of "stations" or "arrays".
Value

A list containing:

• detections: A list containing all detections for each target tag;
• valid.detections: A list containing the valid detections for each target tag;
• spatial: A list containing the spatial information used during the analysis;
• deployments: A data frame containing the deployments of each receiver;
• arrays: A list containing the array details used during the analysis;
• movements: A list containing all movement events for each target tag;
• valid.movements: A list containing the valid movement events for each target tag;
• section.movements: A list containing the valid section shifts for each target tag;
• status.df: A data frame containing summary information for each tag, including the following columns:
  – Times.entered.[section]: Total number of times the tag entered a given section
  – Average.entry.[section]: Average entry time at a given section
  – Average.time.[section]: Average time the tag spent in a given section during each visit
  – Average.departure.[section]: Average departure time from a given section
  – Total.time.[section]: Total time spent in a given section
  – Very.last.array: Last array where the tag was detected
  – Very.last.time: Time of the last valid detection
  – Status: Fate assigned to the animal
  – Valid.detections: Number of valid detections
  – Invalid.detections: Number of invalid detections
  – Valid.events: Number of valid events
  – Invalid.events: Number of invalid events
  – P.type: Type of processing:
    * ‘Skipped’ if no data was found for the tag,
    * ’Auto’ if no user interaction was required,
    * ‘Manual’ if user interaction was suggested and the user made changes to the validity of the events,
    * ‘Overridden’ if the user listed the tag in the override argument.
  – Comments: Comments left by the user during the analysis
• last.seen: A data frame containing the number of tags last seen in each study area section;
• array.times: A data frame containing ALL the entry times of each tag in each array;
• section.times: A data frame containing all the entry times of each tag in each section;
• residency.list: A list containing the places of residency between first and last valid detection for each tag;
• time.ratios: A list containing the daily location per section (both in seconds spent and in percentage of day) for each tag;
• time.positions: A data frame showing the location where each tag spent the most time per day;
• **global.ratios**: A list containing summary tables showing the number of active tag (and respective percentages) present at each location per day;
• **efficiency**: A list containing the results of the inter-array Multi-way efficiency calculations (see vignettes for more details);
• **intra.array.CJS**: A list containing the results of the intra-array CJS calculations;
• **rsp.info**: A list containing appendix information for the RSP package;
• **dist.mat**: The distance matrix used in the analysis (if a valid distance matrix was supplied)

See Also

`explore`, `migration`

Examples

```r
# Start by moving to a temporary directory
old.wd <- getwd()
setwd(tempdir())

# Deploy the example workspace
eexampleWorkspace("exampleWorkspace")

# Move your R session into the example workspace
setwd("exampleWorkspace")

# run the residency analysis. Ensure the tz argument
# matches the time zone of the study area and that the
# sections match your array names. The line below works
# for the example data.
results <- residency(tz = "Europe/Copenhagen")

# to obtain an HTML report, run the analysis with report = TRUE

# return to original working directory
setwd(old.wd)
rm(old.wd)
```

---

**Description**

Computes an analytical CJS model for a presence/absence matrix.

**Usage**

```
simpleCJS(input, estimate = NULL, fixed.efficiency = NULL, silent = TRUE)
```
### stationName

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input</code></td>
<td>A presence/absence matrix.</td>
</tr>
<tr>
<td><code>estimate</code></td>
<td>An estimate of the last array’s efficiency, between 0 and 1.</td>
</tr>
<tr>
<td><code>fixed.efficiency</code></td>
<td>A vector of fixed efficiency estimates [0, 1]. length(fixed.efficiency) must match ncol(input).</td>
</tr>
<tr>
<td><code>silent</code></td>
<td>Logical: Should messages be printed? This argument is mainly intended for function calls running within actel’s analyses.</td>
</tr>
</tbody>
</table>

**Value**

A list containing:

- `absolutes` A data frame with the absolute number of tags detected and missed,
- `efficiency` A vector of calculated array detection efficiencies,
- `survival` A matrix of calculated survivals,
- `lambda` A combined detection efficiency * survival estimate for the last array.

**References**


**Examples**

```r
# prepare a dummy presence/absence matrix
x <- matrix(c(TRUE, TRUE, TRUE, TRUE, FALSE, TRUE, TRUE, TRUE, FALSE), ncol = 3)
colnames(x) <- c("Release", "Array1", "Array2")

# run CJS
simpleCJS(x)
```

---

<table>
<thead>
<tr>
<th>stationName</th>
<th>Find original station name</th>
</tr>
</thead>
</table>

**Description**

Find original station name

**Usage**

```r
stationName(input, station)
```
**Arguments**

- **input**: The results of an actel analysis (either explore, migration or residency).
- **station**: The station standard name or number.

**Value**

The original station name

**Examples**

```r
stationName(example.results, 1)
```

# or

```r
stationName(example.results, "St.2")
```

---

**timesToCircular**  
*Convert a data frame with timestamps into a list of circular objects*

**Description**

Convert a data frame with timestamps into a list of circular objects

**Usage**

```r
timesToCircular(x, by.group = FALSE)
```

**Arguments**

- **x**: A data frame where the first column is an identifier, the second column is a grouping structure, and columns three and onwards are timestamps at different locations. Can be generated automatically by `getTimes`.
- **by.group**: Logical: Should the times at each location be divided by the group column (second column of `x`)?

**Value**

A list of circular objects for each data column and, optionally, for each group.
### transitionLayer

**Calculate Transition Layer**

**Description**

Using a previously imported shape file that has been converted to a raster (see `loadShape`), Prepares a TransitionLayer object to be used in distance estimations (see `distancesMatrix`). Adapted from Grant Adams' script "distance to closest mpa".

**Usage**

```r
transitionLayer(x, directions = c(16, 8, 4))
```

**Arguments**

- `x`: A water raster; for example the output of `loadShape`
- `directions`: The number of directions considered for every movement situation during cost calculation. See the vignettes for more details.

**Details**

It is highly recommended to read the vignette regarding distances matrix before running this function. You can find it by running `vignette('a-2_distances_matrix', 'actel')` or `browseVignettes('actel')`.

---

**Examples**

```r
# create dummy input data frame.
# Note: the names of the columns are irrelevant.
x <- data.frame(ID = c(1:5),
                Group = c("A", "A", "B", "B", "B"),
                A1 = as.POSIXct(c("2019-01-03 11:21:12",
                                "2019-01-05 13:31:34",
                                "2019-01-06 14:32:43",
                                "2019-01-07 15:23:52")),
                A2 = as.POSIXct(c("2019-01-08 16:51:55",
                                "2019-01-09 17:42:42",
                                "2019-01-10 18:33:33",
                                "2019-01-12 20:15:22")),
                stringsAsFactors = TRUE)

# run timesToCircular
timesToCircular(x)

# optionally, split results by group:
timesToCircular(x, by.group = TRUE)
```
Value

A TransitionLayer object.

Examples

```r
# check if R can run the distance functions
aux <- c(
    length(suppressWarnings(packageDescription("raster"))),
    length(suppressWarnings(packageDescription("gdistance"))),
    length(suppressWarnings(packageDescription("sp"))),
    length(suppressWarnings(packageDescription("tools"))),
    length(suppressWarnings(packageDescription("rgdal"))))
missing.packages <- sapply(aux, function(x) x == 1)
if (any(missing.packages)) {
    message("Sorry, this function requires packages ",
            paste(c("raster", "gdistance", "sp", "tools", "rgdal")[missing.packages], collapse = ", "),
            ", to operate. Please install ", ifelse(sum(missing.packages) > 1, "them", "it"),
            ", before proceeding.")
} else {
    if (suppressWarnings(require("rgdal"))) {
        # Fetch actel's example shapefile location
        aux <- system.file(package = "actel")[[1]]

        # import the shape file
        x <- loadShape(path = aux, shape = "example_shapefile.shp", size = 20)

        # Build the transition layer
        t.layer <- transitionLayer(x)

        # inspect the output
        t.layer
    } else {
        message("Sorry, it appears that rgdal is not being able to load.")
    }
}
rm(aux, missing.packages)
```
Index

advEfficiency, 2
completeMatrix, 4
createWorkspace, 5
dataToList, 6
distancesMatrix, 7, 37, 47
dualArrayCJS, 9
eemptyMatrix, 10
eexampleWorkspace, 10
eexplore, 5, 11, 16, 17, 25, 36, 40, 44
eextractCodeSpaces, 14
eextractSignals, 15
getSpeeds, 16
getTimes, 17, 46
loadShape, 18, 47
loadSpatial, 20
migration, 5, 14, 16, 17, 21, 36, 40, 44
plotArray, 26
plotDetections, 27
plotLive, 28
plotMoves, 30
plotRatios, 31
plotResidency, 32
plotSensors, 33
plotTimes, 34
preload, 12, 22, 36, 41
readDot, 38
recoverLog, 39
residency, 5, 14, 16, 17, 25, 36, 40, 40
simpleCJS, 44
stationName, 45
timesToCircular, 46
timezones, 12, 22, 37, 41
transitionLayer, 7, 47