Package: simfish (via r-universe)

August 22, 2024

Type Package

Title simulate fish tracks & acoustic detections

Version 0.2-08.9010

Date 2023-05-13

Author Ian Jonsen

Maintainer Ian Jonsen <ian.jonsen@mq.edu.au>

Description facilitates simulation of fish tracks in featureless and semi-realistic environments (coastal oceans, fjords, and lakes). A user-defined raster defining the water body - land boundaries is used to constrain simulated tracks. Fish movements are simulated (currently) as either a correlated random walk or a biased & correlated random walk, with the bias toward a defined Centre-of-Attraction. A potential function is used to ensure the tracks avoid land. The package can also be used to simulate detections of acoustically-tagged fish by acoustic receivers at user-defined locations.

URL https://github.com/ianjonsen/simfish/,

Depends R (>= 4.2.2)

Imports dplyr, ggplot2, raster (>= 3.6-20), sp, tibble, CircStats, stats, sf (>= 1.0-12), lubridate, stringr, stars, rnaturalearth (>= 0.2.0), methods, pathroutr, terra (>= 1.7-18)

Remotes jmlondon/pathroutr

License AGPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, rnaturalearthhires

VignetteBuilder knitr

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

RemoteUrl https://github.com/mhpob/simfish

RemoteRef HEAD

RemoteSha 94abda3265fd8aa025ee0a0d8e5ace3d92b4214f

Contents

simfish-package   simfish

Description

simulate fish tracks & acoustic detections

Author(s)

Maintainer: Ian Jonsen <ian.jonsen@mq.edu.au> [ORCID](https://orcid.org/0000-0001-5423-6076)

See Also

Useful Links: * https://github.com/ianjonsen/simfish/ * Report bugs/issues at https://github.com/ianjonsen/simfish/issues/

calc_pdrf *calculate pdrf parameters*

Description

A helper function to calculate the logisitc regression intercept and/or slope parameters used to simulate acoustic detection probability as a function of distance between transmitter and receiver

Usage

```
calc_pdrf(
  pdet = NULL,
  dist = NULL,
  int = 5,
  slope = NULL,
  plot = FALSE,
  dmax = 1000,
  dint = 20
)
```

Arguments

`pdet`	probability of detection
`dist`	distance (in m) between transmitter and receiver
`int`	intercept parameter from a logistic regression of distance on p(detection). Default value is 5.
`slope`	slope parameter from a logistic regression of distance on p(detection). Default is NULL.
`plot`	logical (default = FALSE); plot the relationship.
`dmax`	maximum distance to be plotted. Ignored if plot = FALSE.
`dint`	plotting interval between distances. Ignored if plot = FALSE.

Details

The `int` default value (5) is based on empirical data collected from sentinel tags and receivers on the Ocean Tracking Network's Halifax line (https://members.oceantrack.org/OTN/project?ccode=HFX).

Value

the intercept and slope parameters, given `pdet`, `dist`, and one of `int` or `slope`.

Author(s)

Ian Jonsen <`ian.jonsen@mq.edu.au`>

Examples

```
# calculate slope
calc_pdrf(pdet = 0.5, dist = 250, int = 5)

# calculate intercept
calc_pdrf(pdet = 0.5, dist = 250, slope = -0.02, int = NULL)

# calculate slope and plot relationship
calc_pdrf(pdet = 0.5, dist = 250, int = 5, plot = TRUE)
```

`find_route` *find_route*

Description

check for barriers between start and end (CoA) points & find route around barriers via intermediate CoA's to be used by `sim_fish()`. Can be called prior to `sim_fish()` or within `sim_fish()`.

Usage

```
find_route(
  data,
  mpar,
  coas = 12,
  npts = 40,
  land.buff = 2,
  buffer = 5,
  centroids = FALSE,
  keep.coas = NULL
)
```

Arguments

`data`
- a list of required data. If missing then simulation runs on a Cartesian grid (a featureless environment).

`mpar`
- simulation control parameters supplied as a list using `sim_par()`

`coas`
- number of Centres of Attraction to calculate

`npts`
- number of route-finding points to use. A larger number of points can produce smoother routes that do not jump across land, a smaller number of points will produce a less complex route, faster.

`land.buff`
- distance (km) to buffer around land. A small buffer (e.g., 1 - 2 km) can reduce coastline complexity and make route-finding simpler.

`buffer`
- `pathroutr` argument that sets the typical distance from coastline of the visibility graph.

`centroids`

- `pathroutr` argument that turns on/off (default: FALSE) centroids in calculating the visibility graph

`keep.coas`

- index of calculated CoA's to use in the simulation. Default is NULL (use all calculated CoA's).

Value

a list including a matrix of calculated CoA's and a `ggplot2` object visualizing the CoA's. The visualization is useful for determining whether to use all the CoA's or subset them, and for helping to diagnose route-finding failures.

`generate_env` *create environmental raster*

Description

create environmental raster to constraint simulated tracks

Usage

```
generate_env(
  ext = NULL,
  prj = "+proj=merc +datum=WGS84 +units=km",
  res = c(1, 1),
  grad = FALSE,
  dist = 0
)
```

Arguments

`ext` longitude and latitude extents as a 4-element vector: (xmin, ymin, xmax, ymax)

`prj` a crs string or EPSG defining the projection for the raster with units in km. The default projection is a Mercator grid: `prj = "+prj=merc +datum=WGS84 +units=km"`.

`res` resolution of output raster in x,y km (default is 1 km). Smaller values yield higher resolution but longer computation times.

`grad` (logical) generate gradient rasters. The gradient rasters (gradient of distance from water in x and y directions) are required by the potential function used to simulate tracks.

`dist` buffer distance (km) to add to land. This can be used when generating the gradient rasters to keep tracks a specified distance from land. The default is 0 km, no buffer.

Details

the `terra` package is used to generate the land and gradient rasters as SpatRaster objects. This allows higher-resolution rasters to be generated faster than with the `raster` package. The SpatRaster objects are converted to RasterLayer (land barrier) and RasterStack (gradients) objects for use in `sim_fish`.

Value

a raster defining the simulation environment (i.e., ocean with land barriers). Optionally, the gradient rasters, required for the simulation potential function, can also be generated. Only one of the environment raster or the gradient rasters can be returned.

Examples

```
x <- generate_env(ext = c(-70,43,-52,53), res = c(5,5))

raster::plot(x)
```

`map` *map*

Description

a map of single or multiple simulated tracks

Usage

```
map(
  x,
  env = NULL,
  land.poly = TRUE,
  by.id = FALSE,
  coas = FALSE,
  term.pts = TRUE,
  zoom = FALSE,
  downsample = 0,
  ...
)
```

Arguments

`x`	a simfish simulation object
`env`	...
`land.poly`	logical (default: TRUE); should land be displayed using rnaturalearth spatial polygon data, or using the raster used by the simulation. The former will look smoother in higher resolution images.

`by.id` logical (default: FALSE); if x is a multi-track object then colour tracks by id

`coas` logical (default: FALSE); should CoA's be displayed on map

`term.pts` logical (default: TRUE); should terminal (start, end) points be displayed on map

`zoom` logical (default: FALSE); should map be zoomed to extent of simulated track, otherwise map uses extents from the land raster in `env`

`downsample` speed up map rendering by down-sampling the land raster, e.g., `downsample = 1` removes every second pixel. Default is 0 or no down-sampling. Ignored if `land.poly = TRUE`

`...` graphical parameters. Arguments that can be passed to `ggplot2::coord_sf()`, such as `xlim = c(20,40)`.

Value

a ggplot object

Author(s)

Ian Jonsen `<ian.jonsen@mq.edu.au>`

`pdet` *Utility fn to calculate distances b/w transmitter and reciever*

Description

assumes detection is a logistic fn of distance in 3-D

Usage

```
pdet(trs = NULL, rec = NULL, b = NULL, noise = 1)
```

Arguments

`trs` - a tibble of transmission locations (in m) with: id, x, y, et and line id

`rec` - a data.frame of receiver locations (in m) with: x, y, z

`b` - a vector of two parameters (intercept & slope) for logistic detection range function

`noise` - range 0 - 1; simulate effect of noisy environment. Reduces detection prob w dist by specified proportion; default = 1, no reduction

Author(s)

Ian Jonsen `<ian.jonsen@mq.edu.au>`

plot.simfish *plot*

Description

a simple visualization of single or multiple simulated tracks

Usage

```
## S3 method for class 'simfish'
plot(x, by.id = FALSE, ...)
```

Arguments

x	a simfish simulation object
by.id	logical (default: FALSE); if x is a multi-track object then colour tracks by id
...	additional arguments to be ignored

Value

a ggplot object

Author(s)

Ian Jonsen <ian.jonsen@mq.edu.au>

Examples

```
my.par <- sim_par(N = 1440, time.step = 5, start = c(0, 0), coa = c(0,30))

z <- sim_fish(id = 1, mpar = my.par)
plot(z)
```

sim_detect *simulate acoustic transmissions & detection, using* simulate *&* sim_setup *output*

Description

simulates transmissions & detections along simulated track segments within a defined range of acoustic array(s)

Usage

```
sim_detect(s, data, delay = c(50, 130), burst = 5, noise = 1)
```

Arguments

`s` - a simfish class list containing output from sim_fish

`data` - a list containing required environmental data, including receiver locations with x,y,z coordinates, where z is receiver depth.

`delay` - min & max time intervals (s) between transmissions

`burst` - duration of each transmission (s)

`noise` - range 0 - 1; simulate effect of noisy environment. Reduces detection prob w dist by specified proportion; default = 1, no reduction

Author(s)

Ian Jonsen `<ian.jonsen@mq.edu.au>`

`sim_fish` *simulate a fish track*

Description

simulates fish tracks in featureless or semi-realistic environments. Semi-realistic environments are user-defined rasters of water bounded by land, which constrain the simulated tracks. User-supplied acoustic receiver locations can be used to simulate acoustic detections based on a user-defined detection probability with distance. Utility functions are provided to generate water-land rasters (‘generate_land()‘), required gradient rasters (‘generate_grad()‘), and to convert acoustic tag detection range expectations into logistic regression parameters (‘calc_pdrf()‘).

Usage

```
sim_fish(id = 1, data = NULL, mpar = sim_par(), route = TRUE, ...)
```

Arguments

`id` - identifier for simulation run (individual fish)

`data` - a list of required data. If missing then simulation runs on a Cartesian grid (a featureless environment).

`mpar` - simulation control parameters supplied as a list using ‘sim_par()‘ See ‘?sim_par‘ for details on the simulation parameters.

`route` (logical) turns route-finding on or off (default is TRUE, on)

`...` additional, optional arguments to ‘find_route()‘

Examples

```
## A minimal example - simulation with no environment
my.par <- sim_par(N = 500, time.step = 10, start = c(0, 0), coa = c(0,30))

z <- sim_fish(id = 1, mpar = my.par)

plot(z)

## Simulate in a semi-realistic environment
x <- generate_env(ext = c(-70,43,-52,53), res = c(5,5), grad = TRUE)

my.par <- sim_par(N=400, time.step=60*6, start = c(-7260, 5930),
coa = c(-6300, 6680), nu = 0.6, rho = 0.7)
z <- sim_fish(id = 1, data = x, mpar = my.par)

map(z, env = x)
```

---

sim_par *Define the simulation parameters used by* sim_fish

Description

Define simulation parameters

Usage

```
sim_par(...)
```

Arguments

... simulation parameters:

N - number of time steps to simulate, if NULL then simulation runs until fish reaches the tolerance distance to the final Centre-of-Attraction (coa.tol);

time.step - duration (min) of each simulation time step;

start.dt - start datetime (POSIXct) of simulation;

start - start location coordinates (grid x, y in km);

coa - Centre-of-Attraction coordinates (grid x, y in km). If > 1 coa supplied (as a matrix) then the simulation uses the coa's sequentially - for example, to guide the track around land features lying between the specified start and end points;

coa.tol - tolerance distance (km) around CoA(s);

nu - strength of attraction to CoA (range: 0 - infinity);

rho - angular dispersion parameter (Wrapped Cauchy) for turn angles;

bearing - a scalar or vector of bearings (in radians) that give the bias in bcrw directions. Ignored if model = bcrw.coa

bl - move speed in body-lengths per second. Can be a vector, if model = "bcrw". If bl is a vector then it must have the same length as bearing;

fl - fish fork-length (m);

pdrf - logistic intercept & slope parameters defining detection probability as a function of distance between fish and receiver. Default values roughly correspond to detection range of V9 tags in the ocean;

beta - potential function parameters for x and y directions to keep fish off land. Larger -ve values result in stronger land avoidance but can introduce unrealistic jumps (possibly across narrow land features) in the track;

buffer - distance radius in km to search for water when simulation stuck on land. Smaller values reduce chance of unrealistic jumps away from land but can lead to simulations getting stuck on land. Larger values increase chance of unrealistic jumps (possibly across land) but reduce chance of simulations getting stuck on land.

Value

Returns a list of simulation parameters to be used by sim_fish. See example, below

Examples

```
## A minimal example - simulation with no environment
my.par <- sim_par(N = 1440, time.step = 5, start = c(0, 0), coa = c(0,30))

z <- sim_fish(id = 1, mpar = my.par)

plot(z)
```

Index