Package 'dsims'

August 30, 2022

Depends dssd (>= 0.2.2)

Imports mrds, Distance, sf, sp, ggplot2, mgcv, rgeos, methods, rstudioapi, gridExtra, rlang

Suggests testthat, parallel, pbapply, knitr, lwgeom, rmarkdown

VignetteBuilder knitr

Type Package

Title Distance Sampling Simulations

Version 1.0.1

Maintainer Laura Marshall <lhm@st-and.ac.uk>

Description Performs distance sampling simulations. 'dsims' repeatedly generates instances of a user defined population within a given survey region. It then generates realisations of a survey design and simulates the detection process. The data are then analysed so that the results can be compared for accuracy and precision across all replications. This process allows users to optimise survey designs for their specific set of survey conditions. The effects of uncertainty in population distribution or parameters can be investigated under a number of simulations so that users can be confident that they have achieved a robust survey design before deploying vessels into the field. The distance sampling designs used in this package from 'dssd' are detailed in Chapter 7 of Advanced Distance Sampling, Buckland et. al. (2008, ISBN-13: 978-0199225873). General distance sampling methods are detailed in Introduction to Distance Sampling: Estimating Abundance of Biological Populations, Buckland et. al. (2004, ISBN-13: 978-0198509271). Find out more about estimating animal/plant abundance with distance sampling at <http://distancesampling.org/>.

License GPL (>= 2)

Language en-GB

URL https://github.com/DistanceDevelopment/dsims

BugReports https://github.com/DistanceDevelopment/dsims/issues

Encoding UTF-8

RoxygenNote 7.2.1

Collate 'AICc.R' 'generic.functions.R' 'Density.Summary.R' 'Density.R'
'Population.Description.R' 'Detectability.R' 'Population.R'
'Survey.R' 'DS.Analysis.R' 'Simulation.R' 'ClassConstructors.R'
'Simulation.Summary.R' 'Survey.LT.R' 'Survey.PT.R'
'accumulate.PP.results.R' 'accumulate.warnings.R'
'add.covariate.values.R' 'add.summary.results.R'
'calc.perp.dists.R' 'calc.rad.dists.R'
'calculate.scale.param.R' 'check.covariates.R'
'check.simulation.R' 'check.transects.R'
'create.results.arrays.R' 'description.summary.R'
'dsims-package.R' 'generate.pop.D.R' 'generate.pop.N.R'
'get.covered.area.lines.R' 'get.covered.area.points.R'
'get.density.surface.R' 'message.handler.R'
'modify.strata.for.analysis.R' 'process.dist.shapes.R'
'read.line.transects.R' 'read.point.transects.R'
'read.seg.transects.R' 'run.simulation.R' 'rztpois.R'
'save.sim.results.R' 'simulate.detections.R'
'single.sim.loop.R' 'store.ddf.results.R' 'store.dht.results.R'

NeedsCompilation no

Author Laura Marshall [aut, cre],
Thomas Len [ctb]

Repository CRAN

Date/Publication 2022-08-30 15:10:06 UTC

R topics documented:

`dsims-package` *Distance Sampling Simulations 'dsims'*

Description

Runs simulations of distance sampling surveys to help users optimise their survey designs for their particular study.

Details

The full process involves defining the study region, a description of the population of interest (including its distribution within the study region), a survey design, a detection process and one or more models to fit to the resulting data. The simulation engine will then use this information to generate both a population and a set of transects and simulate the detection process. The resulting data will be analysed and the estimates stored. By repeating this many times we can test the accuracy and precision of our estimates from various survey designs given our particular population of interest.

This package interfaces with the survey design package 'dssd' to create the survey regions, designs and generate the survey transects. While the 'DSsim' simulation package relied on survey transects already being contained in shapefiles within the supplied directory, dsims will generate the survey transects directly in R.

The main functions in this package are: make.density, make.population.description, make.detectability, make.ds.analysis, make.simulation, run.survey and run.simulation. See also make.region and make.design in the dssd package for examples of how to define study regions and designs.

Further information on distance sampling methods and example code is available at http://distancesampling.org/R/.

We are also in the process of setting up a new area of the website for vignettes / example code at http://examples.distancesampling.org. While this is being developed, the 'dsims' vignette can still be found within this package.

For help with distance sampling and this package, there is a Google Group https://groups.google.com/forum/#!forum/distance-sampling.

Author(s)

Laura Marshall <lhm@st-and.ac.uk>

`add.hotspot` *S4 generic method to add a hotspot to the density grid*

Description

Uses a Gaussian decay around a central location to add a hotspot to the density grid.

Usage

```
add.hotspot(object, centre, sigma, amplitude)

## S4 method for signature 'Density'
add.hotspot(object, centre, sigma, amplitude)
```

Arguments

`object`	a `Density-class` object
`centre`	an x,y-coordinate giving the centre of the hotspot
`sigma`	a value giving the scale parameter for a gaussian decay
`amplitude`	the height of the hotspot at its centre

Value

the updated `Density-class` object

See Also

`make.density`

`analyse.data` *S4 generic method to run analyses*

Description

This method carries out an analysis of distance sampling data. This method is provided to allow the user to perform diagnostics of the analyses used in the simulation. The data argument can be obtained by a call to `simulate.survey(object, dht.table = TRUE)`. Note if the first object supplied is of class DS.Analysis then the second argument must be of class DDf.Data. The data argument may be of either class for an object argument of class Simulation.

Usage

```
analyse.data(analysis, data.obj, ...)

## S4 method for signature 'DS.Analysis,Survey'
analyse.data(analysis, data.obj, warnings = NULL, ...)

## S4 method for signature 'DS.Analysis,data.frame'
analyse.data(analysis, data.obj, warnings = NULL, transect = "line", ...)
```

Arguments

`analysis`	an object of class DS.Analysis
`data.obj`	an object of class Survey or a dataframe
`...`	optional arguments (currently not used)
`warnings`	a list of warnings and how many times they arose
`transect`	character value either "line" or "point" specifying type of transect used in survey

Value

a list containing an S3 ddf object and optionally an S3 dht object relating to the model with the minimum criteria.

either returns a list of the best model, warnings and the number of successfully fitted models (if warnings is supplied as a list) otherwise displays warnings as it goes and returns the best fitting ds model.

Density-class *Class "Density"*

Description

Class "Density" is an S4 class containing a list of grids which describe the density of individuals / clusters of a population. The list contains one grid (data.frame) for each strata.

Slots

region.name Object of class "character"; the region name.

strata.name Object of class "character"; the strata names

density.surface Object of class "list"; list of data.frames with the columns x, y and density. There must be one data.frame for each strata.

x.space Object of class "numeric"; The spacing between gridpoints described in the density data.frames in the x-direction.

y.space Object of class "numeric"; The spacing between gridpoints described in the density data.frames in the y-direction.

units Object of class "numeric"; The units of the grid points.

See Also

make.density

Density.Summary-class *Class "Density.Summary"*

Description

Class "Density.Summary" is an S4 class containing a summary of the density grids for each strata.

Slots

summary a summary of the average abundances and densities for each strata.

See Also

make.density

description.summary *Provides a description of the summary object/output*

Description

Prints a list of the terms used in the simulation summary.

Usage

```
description.summary()
```

Value

no return, displays an explanation of the simulation summary

Author(s)

Laura Marshall

Detectability-class *S4 Class "Detectability"*

Description

S4 Class "Detectability"

Slots

`key.function` Object of class `"character"`; a code specifying the detection function form ("hn" = half normal, "hr" = hazard rate.)

`scale.param` Object of class `"numeric"`; The scale parameter for the detection function.

`shape.param` Object of class `"numeric"`; The shape parameter for the detection function.

`cov.param` Object of class `"numeric"`; The parameter values associated with the covariates. Not yet implemented

`truncation` Object of class `"numeric"`; The maximum distance at which objects may be detected.

See Also

`make.detectability`

DS.Analysis-class *Class "DS.Analysis"*

Description

Class "DDF.Analysis" is an S4 class describing a basic detection function model to be fitted to distance sampling data.

Slots

dfmodel Object of class "formula"; describing the detection function model.

key key function to use; "hn" gives half-normal (default), "hr" gives hazard-rate and "unif" gives uniform. Note that if uniform key is used, covariates cannot be included in the model.

adjustment a list containing adjustment parameters: adjustment - either "cos" (recommended), "herm" or "poly", order - the orders of the adjustment terms to fit, scale - the scale by which the distances in the adjustment terms are divided. See details.

truncation Object of class "list"; Specifies the truncation distance for the analyses.

cutpoints Object of class "character"; gives the cutpoints of the bins for binned data analysis.

er.var specifies which encounter rate variance estimator to use.

control.opts A list to specify various options including monotonicity, method, initial.values.

group.strata Dataframe with two columns ("design.id" and "analysis.id"). The former gives the strata names as defined in the design (i.e. the region object) the second specifies how they should be grouped (into less strata) for the analyses

criteria Object of class "character"; describes which model selection criteria to use ("AIC","AICc","BIC").

Methods

run.analysis signature=c(object = "DS.Analysis", data = data.frame): runs the analysis described in the object on the data provided.

generate.population *S4 generic method to generate an instance of a population*

Description

Uses the population description and detectability details to generate an instance of the population. Note that if the first argument supplied is of class Population.Description rather than class Simulation then detectability and region must also be supplied.

Usage

```
generate.population(object, ...)

## S4 method for signature 'Population.Description'
generate.population(object, detectability = NULL, region = NULL)

## S4 method for signature 'Simulation'
generate.population(object, ...)
```

Arguments

object	an object of class Simulation or Population.Description
...	when this is called on an object of class Population.Description the additional arguments detectability and region.obj should also be supplied
detectability	object of class Detectability (optional - only required if object is of class Population.Description)
region	the region object for the population (optional - only required if object is of class Population.Description)

Value

Population-class object

---

generate.transects,Simulation-method
generate.transects

Description

Generates a set of transects based on the design provided.

Usage

```
## S4 method for signature 'Simulation'
generate.transects(object, quiet = FALSE, ...)
```

Arguments

object	object of class Simulation
quiet	if TRUE silences some warnings
...	not implemented

Value

an object of class Transect from dssd package

get.densities *Method to get density values*

Description

This method extracts the density values from a density object. It will optionally also return the x and y centre points for the density grid cells.

Usage

```
get.densities(density, coords = FALSE)
```

Arguments

density	object of class Density
coords	if TRUE also returns x, y coordinates

Value

either returns a numeric vector of density values or a dataframe with columns x, y and density.

get.N *S4 generic method to return N*

Description

Returns the population size

Usage

```
get.N(object)

## S4 method for signature 'Population.Description'
get.N(object)
```

Arguments

object	an object of class Population.Description

Value

numeric value of the population size

histogram.N.ests *histogram.N.ests*

Description

Plots a histogram of the estimates abundances

Usage

```
histogram.N.ests(x, use.max.reps = FALSE, N.ests = "individuals", ...)
```

Arguments

x	object of class Simulation
use.max.reps	by default this is FALSE meaning that only simulation repetitions where all models converged for that data set are included. By setting this to TRUE any repetition where one or more models converged will be included in the summary results.
N.ests	character indicating whether to plot estimates of abundance of 'individuals', 'clusters' or 'both'. By default this is individuals.
...	optional parameters to pass to the generic hist function in graphics

Value

No return value, displays a histogram of the abundance estimates

make.density *Creates a Density object*

Description

Creates a density grid across the study area describing the distribution of animals.

Usage

```
make.density(
  region = make.region(),
  x.space = 20,
  y.space = NULL,
  constant = numeric(0),
  fitted.model = NULL,
  density.formula = NULL,
  density.surface = list()
)
```

Arguments

region	the Region object in which the density grid will be created
x.space	the intervals in the grid in the x direction
y.space	the intervals in the grid in the y direction
constant	a value describing a constant density across the surface. If not supplied a default value of 1 is used for all strata.
fitted.model	gam object created using mgcv with only x and y as explanatory covariates.
density.formula	a formula of x and/or y describing the density surface.
density.surface	Object of class list; an sf grid recording the density grid polygons, density values within those polygons and the central x and y coordinates.

Details

There are multiple ways to create the density grid. The most straight forward is to create a grid with constant values (to which high and low areas can later be added) or pass in a fitted mgcv gam. The gam model should only be fitted with x and y as explanatory variables. If you plan on trying multiple animal distributions by adding high and low areas to a constant surface it is recommended to make a copy of the initial flat density grid object as the first step in grid generation is computationally intensive and can take a little while to complete, especially if you have a fine density grid.

Value

Density-class object

Author(s)

Laura Marshall

See Also

make.region

Examples

```
# A simple density surface with a constant value of 1 can be created within a rectangular
# Create a region from shapefile
shapefile.name <- system.file("extdata", "StAndrew.shp", package = "dssd")
region <- make.region(region.name = "St Andrews bay",
                      shape = shapefile.name)

# Create a density object
density <- make.density(region = region,
                        x.space = 1000,
                        constant = 1)

# Add some ares of higher / lower density
```

```
density <- add.hotspot(object = density,
                       centre = c(-170000, 6255000),
                       sigma = 10000,
                       amplitude = 4)
density <- add.hotspot(object = density,
                       centre = c(-150000, 6240000),
                       sigma = 10000,
                       amplitude = -0.9)

# Plot the density
plot(density, region)
```

`make.detectability` *Creates a Detectability object*

Description

The detectability of the population is described by the values in this class.

Usage

```
make.detectability(
  key.function = "hn",
  scale.param = 25,
  shape.param = numeric(0),
  cov.param = list(),
  truncation = 50
)
```

Arguments

`key.function`	specifies shape of the detection function (either half-normal "hn", hazard rate "hr" or uniform "uf")
`scale.param`	numeric vector with either a single value to be applied globally or a value for each strata. These should be supplied on the natural scale.
`shape.param`	numeric vector with either a single value to be applied globally or a value for each strata. These should be supplied on the natural scale.
`cov.param`	Named list with one named entry per individual level covariate. Covariate parameter values should be defined on the log scale (rather than the natural scale), this is the same scale as provided in the ddf output in mrds and also in the MCDS output in Distance. Cluster sizes parameter values can be defined here. Each list entry will either be a data.frame containing 2 or 3 columns: level, param and where desired strata. If the region has multiple strata but this column is omitted then the values will be assumed to apply globally. The cluster size entry in the list must be named 'size'. Alternatively the list element may a numeric vector with either a single value to be applied globally or a value for each strata.
`truncation`	the maximum perpendicular (or radial) distance at which objects may be detected from a line (or point) transect.

Value

Detectability-class object

Author(s)

Laura Marshall

See Also

make.simulation make.population.description make.density

Examples

```
# Multi-strata example (make sf shape)
s1 = matrix(c(0,0,0,2,1,2,1,0,0,0),ncol=2, byrow=TRUE)
s2 = matrix(c(1,0,1,2,2,2,2,0,1,0),ncol=2, byrow=TRUE)
pol1 = sf::st_polygon(list(s1))
pol2 = sf::st_polygon(list(s2))
sfc <- sf::st_sfc(pol1,pol2)
strata.names <- c("low", "high")
sf.pol <- sf::st_sf(strata = strata.names, geom = sfc)

region <- make.region(region.name = "Multi-strata Eg",
                      strata.name = strata.names,
                      shape = sf.pol)

density <- make.density(region = region,
                        x.space = 0.22,
                        constant = c(20,50))

covs <- list()
covs$size <- list(list(distribution = "poisson", lambda = 25),
                  list(distribution = "poisson", lambda = 15))
covs$sex <- data.frame(level = rep(c("male", "female"),2),
                      prob = c(0.5, 0.5, 0.6, 0.4),
                      strata = c(rep("low",2),rep("high",2)))

# Define the population description (this time using the density to determine
# the population size)
popdesc <- make.population.description(region = region,
                                       density = density,
                                       covariates = covs,
                                       fixed.N = FALSE)

cov.param <- list()
cov.param$size <- c(log(1.02),log(1.005))
cov.param$sex <- data.frame(level = c("male", "female", "male", "female"),
                            param = c(log(1.5), 0, log(1.7), log(1.2)),
                            strata = c("low","low","high","high"))

# define the detecability
detect <- make.detectability(key.function = "hn",
```

```
                              scale.param = 0.08,
                              cov.param = cov.param,
                              truncation = 0.2)

plot(detect, popdesc)
```

`make.ds.analysis` *Creates an Analysis object*

Description

This method creates an Analysis objects which describes a one or more models to fit to the distance data. The simulation will fit each of these models to the data generated in the simulation and select the model with the minimum criteria value.

Usage

```
make.ds.analysis(
  dfmodel = list(~1),
  key = "hn",
  truncation = numeric(0),
  cutpoints = numeric(0),
  er.var = "R2",
  control.opts = list(),
  group.strata = data.frame(),
  criteria = "AIC"
)
```

Arguments

`dfmodel`	list of distance sampling model formula specifying the detection function (see `?Distance::ds` for further details)
`key`	key function to use; "hn" gives half-normal (default) and "hr" gives hazard-rate.
`truncation`	absolute truncation distance in simulation units matching the region units.
`cutpoints`	supply a vector of cutpoints if you wish the simulation to perform binned analyses.
`er.var`	encounter rate variance estimator to use when abundance estimates are required. Defaults to "R2" for line transects and "P3" for point transects. See `mrds::varn` for more information / options.
`control.opts`	A list of control options: method - optimisation method,
`group.strata`	Dataframe with two columns ("design.id" and "analysis.id"). The former gives the strata names as defined in the design (i.e. the region object) the second specifies how they should be grouped (into less strata) for the analyses. See details for more information.
`criteria`	character model selection criteria (AIC, AICc, BIC)

Details

It is possible to group strata at the analysis stage using the group.strata argument. For example, for design purposes it may have been sensible to divide strata into substrata. This can help make more convex shapes and therefore zigzag designs more efficient or perhaps it helped to keep transects angled parallel to density gradients across the study area. Despite these (purely design relevant) substrata we may still wish to calculate estimates of density / abundance etc. for each stratum. The table below gives an example of the data.frame which can be used to do this. Imagine a study region with an onshore strata and an offshore strata. The onshore strata has been divided in two at the design stage to keep transects perpendicular to the coast. We now want to analyse this as just two strata the onshore and offshore.

design.id	analysis.id
onshoreN	onshore
onshoreS	onshore
offshore	offshore

Value

DS.Analysis-class object

Author(s)

Laura Marshall

See Also

ds make.simulation

Examples

```
# Model selection considering both a half-normal and a hazard-rate model
# using AIC criteria and truncating 5% of the data
ds.analyses <- make.ds.analysis(dfmodel = ~1,
                                key = c("hn", "hr"),
                                truncation = 500,
                                criteria = "AIC")

# Model selection considering both a half-normal with no covariates and with size
# as a covariate using AIC criteria and truncating at 500
ds.analyses <- make.ds.analysis(dfmodel = list(~1, ~size),
                                key = "hn",
                                truncation = 500,
                                criteria = "AIC")

# Model selection considering both a half-normal with no covariates and with size
# as a covariate and a hazard rate, using AIC criteria and truncating at 500
ds.analyses <- make.ds.analysis(dfmodel = list(~1, ~size, ~1),
```

```
                                key = c("hn", "hn", "hr"),
                                truncation = 500,
                                criteria = "AIC")
```

`make.population.description` *Creates a Population.Description object*

Description

Creates an object which describes a population. The values in this object will be used to create instances of the population.

Usage

```
make.population.description(
  region = make.region(),
  density = make.density(),
  covariates = list(),
  N = numeric(0),
  fixed.N = TRUE
)
```

Arguments

`region`	the Region object in which this population exists (see make.region).
`density`	the Density object describing the distribution of the individuals / clusters (see make.density).
`covariates`	Named list with one named entry per individual-level covariate. Cluster sizes can be defined here, it must be named 'size'. The distribution of covariate values can either be defined by specifying a particular distribution and its parameters or as a discrete distribution in a dataframe. Dataframes should have columns level and prob (and optionally strata) specifying the covariates levels, probabilities and strata if they are strata specific. Distributions can be defined as lists with named entries distribution and the relevant parameters as specified in details. A list of distributions can be provided with one for each strata.
`N`	the number of individuals / clusters in a population with one value per strata. Total population size is 1000 by default.
`fixed.N`	a logical value. If TRUE the population is generated from the value(s) of N otherwise it is generated from the values in the density grid.

Details

Individual-level covariate values can be defined as one of the following distributions: 'normal', 'poisson', 'ztruncpois' or 'lognormal'. The distribution name and the associated parameters as defined in the table below must be provided in a named list. Either one list can be provided for the entire study area or multiple lists grouped together as a list with one per strata.

Distribution	Parameters	
normal	mean	sd
poisson	lambda	
ztruncpois	mean	
lognormal	meanlog	sdlog

Value

Population.Description-class

Author(s)

Laura Marshall

See Also

make.simulation make.detectability make.density

Examples

```
# Create a basic rectangular study area
region <- make.region()

# Make a density grid (large spacing for speed)
density <- make.density(region = region,
                        x.space = 200,
                        y.space = 100,
                        constant = 1)
density <- add.hotspot(density, centre = c(1000, 100), sigma = 250, amplitude = 10)

# Define some covariate values for out population
covs <- list()
covs$size <- list(distribution = "ztruncpois", mean = 5)

# Define the population description
popdsc <- make.population.description(region = region,
                                      density = density,
                                      covariates = covs,
                                      N = 200)

# define the detecability
detect <- make.detectability(key.function = "hn", scale.param = 25, truncation = 50)

# generate an example population
pop <- generate.population(popdsc, region = region, detectability = detect)

plot(pop, region)

# Multi-strata example (make sf shape)
s1 = matrix(c(0,0,0,2,1,2,1,0,0,0),ncol=2, byrow=TRUE)
s2 = matrix(c(1,0,1,2,2,2,2,0,1,0),ncol=2, byrow=TRUE)
```

```
pol1 = sf::st_polygon(list(s1))
pol2 = sf::st_polygon(list(s2))
sfc <- sf::st_sfc(pol1,pol2)
strata.names <- c("low", "high")
sf.pol <- sf::st_sf(strata = strata.names, geom = sfc)

region <- make.region(region.name = "Multi-strata Eg",
                      strata.name = strata.names,
                      shape = sf.pol)

density <- make.density(region = region,
                        x.space = 0.22,
                        constant = c(10,80))

covs <- list()
covs$size <- list(list(distribution = "poisson", lambda = 25),
                  list(distribution = "poisson", lambda = 15))
covs$sex <- data.frame(level = rep(c("male", "female"),2),
                      prob = c(0.5, 0.5, 0.6, 0.4),
                      strata = c(rep("low",2),rep("high",2)))

# Define the population description (this time using the density to determine
# the population size)
popdesc <- make.population.description(region = region,
                                       density = density,
                                       covariates = covs,
                                       fixed.N = FALSE)

# define the detecability (see make.detectability to alter detection function
# for different covariate values)
detect <- make.detectability(key.function = "hn", scale.param = 25, truncation = 50)

# generate an example population
pop <- generate.population(popdesc, region = region, detectability = detect)

plot(pop, region)
```

`make.simulation` *Creates a Simulation object*

Description

This creates a simulation with all the information necessary for dsims to generate a population, create transects, simulate the survey process and fit detection functions and estimate density / abundance. This function can be used by itself based on default values to create a simple line transect example, see Examples below. To create more complex simulations it is advisable to define the different parts of the simulation individually before grouping them together. See the Arguments for links to the functions which make the definitions for the individual simulation components. For a more in depth example please refer to the 'GettingStarted' vignette.

Usage

```
make.simulation(
  reps = 10,
  design = make.design(),
  population.description = make.population.description(),
  detectability = make.detectability(),
  ds.analysis = make.ds.analysis()
)
```

Arguments

reps	number of times the simulation should be repeated
design	an object of class Survey.Design created by a call to make.design
population.description	an object of class Population.Description created by a call to make.population.description
detectability	and object of class Detectability created by a call to make.detectability
ds.analysis	an objects of class DS.Analysis created by a call to make.ds.analysis

Details

The make.simulation function is now set up so that by default (with the exception of specifying point transects rather than line) it can run a simple simulation example. See examples.

Value

Simulation-class object

Author(s)

Laura Marshall

See Also

make.region make.density make.population.description make.detectability make.ds.analysis make.design

Examples

```
# Create a basic rectangular study area
region <- make.region()

# Make a density grid (large spacing for speed)
density <- make.density(region = region,
                        x.space = 300,
                        y.space = 100,
                        constant = 1)
density <- add.hotspot(density, centre = c(1000, 100), sigma = 250, amplitude = 10)

# Define the population description
```

```
popdsc <- make.population.description(region = region,
                                      density = density,
                                      N = 200)

# Define the detecability
detect <- make.detectability(key.function = "hn",
                             scale.param = 25,
                             truncation = 50)

# Define the design
design <- make.design(region = region,
                      transect.type = "line",
                      design = "systematic",
                      samplers = 20,
                      design.angle = 0,
                      truncation = 50)

# Define the analyses
ds.analyses <- make.ds.analysis(dfmodel = ~1,
                                key = "hn",
                                truncation = 50,
                                criteria = "AIC")

# Put all the components together in the simulation (note no. of replicates
# reps = 1 is only for a single test run and should be 999 or more to be able
# to draw inference.)
simulation <- make.simulation(reps = 1,
                              design = design,
                              population.description = popdsc,
                              detectability = detect,
                              ds.analysis = ds.analyses)

# run an example survey to check the setup
survey <- run.survey(simulation)
plot(survey, region)

# Run the simulation
# Warning: if you have increased the number of replications then it can take a
# long time to run!
simulation <- run.simulation(simulation)
summary(simulation)

# For a more in depth example please look at
vignette("GettingStarted", 'dsims')
```

---

plot,Density,ANY-method

Plot

Description

Plots an S4 object of class 'Density'

Plots an S4 object of class 'Density'

Usage

```
## S4 method for signature 'Density,ANY'
plot(x, y, strata = "all", title = "", scale = 1)

## S4 method for signature 'Density,Region'
plot(x, y, strata = "all", title = "", scale = 1, line.col = gray(0.2))
```

Arguments

x	object of class Density
y	object of class Region
strata	the strata name or number to be plotted. By default all strata will be plotted.
title	plot title
scale	used to scale the x and y values in the plot (warning may give unstable results when a projection is defined for the study area!)
line.col	sets the line colour for the shapefile

Value

ggplot object

ggplot object

`plot,Detectability,ANY-method`
Plot

Description

Plots an S4 object of class 'Detectability'

Usage

```
## S4 method for signature 'Detectability,ANY'
plot(
  x,
  y,
  add = FALSE,
  plot.units = character(0),
  region.col = NULL,
  gap.col = NULL,
```

```
  main = "",
  ...
)

## S4 method for signature 'Detectability,Population.Description'
plot(
  x,
  y,
  add = FALSE,
  plot.units = character(0),
  region.col = NULL,
  gap.col = NULL,
  main = "",
  ...
)
```

Arguments

x	object of class Detectability
y	object of class Population.Description
add	logical indicating whether it should be added to existing plot
plot.units	allows for units to be converted between m and km
region.col	fill colour for the region
gap.col	fill colour for the gaps
main	character plot title
...	other general plot parameters

Value

No return value, gives a warning to the user

No return value, plotting function

`plot,Population,ANY-method` *Plot*

Description

Unused, will give a warning that the region must also be supplied.

Plots an S4 object of class 'Population'. Requires that the associated region has already been plotted. This function adds the locations of the individuals/clusters in the population.

Usage

```
## S4 method for signature 'Population,ANY'
plot(x, y, ...)

## S4 method for signature 'Population,Region'
plot(x, y, ...)
```

Arguments

x	object of class Population
y	object of class Region
...	other general plot parameters

Value

ggplot object

---

plot,Survey,Region-method
plot

Description

Produces four plots of the survey: 1) Plots the transects inside the survey region, 2) plots the population, 3) plots the transects, population and detections 4) plots a histogram of the detection distances. Note that only plots 3 & 4 are generated without the survey region if Region is omitted.

Usage

```
## S4 method for signature 'Survey,Region'
plot(x, y, type = "all", ...)

## S4 method for signature 'Survey,ANY'
plot(x, y = NULL, type = "all", ...)
```

Arguments

x	object of class Survey
y	object of class Region or NULL
type	character specifies which plots you would like, defaults to "all". Other options include "transects", "population", "survey" and "distances". These will plot only the transects, only the population locations, both the transects and population with detections indicated or a histogram of the detection distances, respectively. Note that the final plots is only available if there were one or more detections. Only "survey" and "distances" available if the y Region argument is not supplied.
...	additional plotting parameters

Value

Generate 4 plots showing the survey population, transects (including covered areas), detections and a histogram of the detection distances. Plots include the survey region. Also invisibly returns a list of ggplot objects if the user would like to customise the plots.

Generate 2 plots showing the survey population, transects (including covered areas), detections and a histogram of the detection distances. Plots do not include survey region. Also invisibly returns a list of ggplot objects if the user would like to customise the plots.

Population-class *Class "Population"*

Description

Contains an instance of a population including a description of their detectability in the form of an object of class Detectability.

Slots

`region.name` Object of class `"character"`; the name of the region object.

`strata.names` Object of class `"character"`; the names of the strata.

`N` Object of class `"numeric"`; the number of individuals/clusters.

`D` Object of class `"numeric"`; the density of individuals/clusters.

`population` Object of class `"data.frame"`; the locations of individuals/clusters and any population covariates.

`detectability` Object of class `"Detectability"`; describes how easily the individuals/clusters can be detected.

Methods

`plot signature=(object = "Line.Transect")`: plots the locations of the individuals/clusters.

See Also

`make.population.description`, `make.detectability`

Population.Description-class *Class "Population.Description"*

Description

Class "Population.Description" is an S4 class containing a description of the population. It provides methods to generate an example population.

Slots

N Object of class "numeric"; number of individuals in the population (optional).

density Object of class "Density"; describes the population density

region.name Object of class "character"; name of the region in which the population exists.

strata.names Character vector giving the strata names for the study region.

covariates Named list with one named entry per individual level covariate. Cluster sizes can be defined here. Each list entry will either be a data.frame containing 2 columns, the first the level (level) and the second the probability

size logical value indicating whether the population occurs in clusters. (prob). The cluster size entry in the list must be named 'size'.

gen.by.N Object of class "logical"; If TRUE N is fixed otherwise it is generated from a Poisson distribution.

Methods

get.N signature=(object = "Population.Description"): returns the value of N

generate.population signature=(object = "Population.Description"): generates a single realisation of the population.

See Also

make.population.description

run.simulation *Method to run a simulation*

Description

Runs the simulation and returns the simulation object with results. If running in parallel and max.cores is not specified it will default to using one less than the number of cores / threads on your machine. For example code see make.simulation

Usage

```
run.simulation(
  simulation,
  run.parallel = FALSE,
  max.cores = NA,
  counter = TRUE,
  transect.path = character(0),
  progress.file = character(0)
)
```

Arguments

simulation	Simulation-class object
run.parallel	logical option to use multiple processors
max.cores	integer maximum number of cores to use, if not specified then one less than the number available will be used.
counter	logical indicates if you would like to see the progress counter.
transect.path	character gives the pathway to a folder of shapefiles or the path to a single shapefile (.shp file) which give the transects which should be used for the simulations. If a folder of transects a new shapefile will be used for each repetition. If a path specifying a single shapefile then the same transects will be used for each repetition.
progress.file	character path with filename to output progress to file for Distance for Windows progress counter. Not to be used when running directly in R.

Value

the Simulation-class object which now includes the results

See Also

make.simulation

---

run.survey *S4 generic method to simulate a survey*

Description

Simulates the process by which individuals or clusters are detected. If a simulation is passed in then it will generate a population, set of transects and simulate the detection process. If a survey is passed in it will simply simulate the detection process. See make.simulation for example usage.

Usage

```
run.survey(object, ...)

## S4 method for signature 'Simulation'
run.survey(object, filename = character(0))

## S4 method for signature 'Survey.LT'
run.survey(object, region = NULL)

## S4 method for signature 'Survey.PT'
run.survey(object, region = NULL)
```

Arguments

object	an object of class Simulation
...	allows extra arguments
filename	optional argument specifying a path to a shapefile if the transects are to be loaded from file.
region	an object of class Region.

Value

An object which inherits from a Survey-class object. This will be a Survey.LT-class object in the case of a simulation with a line transect design and a Survey.PT-class if the simulation has a point transect design.

See Also

make.simulation

---

rztpois *Randomly generates values from a zero-truncated Poisson distribution*

Description

Generates values from a zero-truncated Poisson distribution with mean equal to that specified. It uses an optimisation routine to check which value of lambda will give values with the requested mean.

Usage

```
rztpois(n, mean = NA)
```

Arguments

n	number of values to randomly generate
mean	mean of the generated values

Value

returns a randomly generated value from a zero-truncated Poisson distribution.

Note

Internal function not intended to be called by user.

Author(s)

Len Thomas

`save.sim.results` *save.sim.results*

Description

Saves the simulation results from each replicate to file. It will save up to 3 txt files, one for the abundance estimation for individuals, one for the abundance estimation of clusters (where applicable) and one for detectability estimates and model selection information.

Usage

```
save.sim.results(simulation, filepath = character(0), sim.ID = numeric(0))
```

Arguments

`simulation`	object of class `Simulation` which has been run.
`filepath`	optionally a path to the directory where you would like the files saved, otherwise it will save it to the working directory.
`sim.ID`	optionally you can add a simulation ID to the filename

Value

invisibly returns the original simulation object

Author(s)

L. Marshall

set.densities *Method to set density values*

Description

This method sets the density values in a density object.

Usage

```
set.densities(density, densities)
```

Arguments

density	object of class Density
densities	a numeric vector of density values to update the density grid with.

Value

returns the Density object with updated density values

show,Density.Summary-method *show*

Description

displays the density summary table

Usage

```
## S4 method for signature 'Density.Summary'
show(object)
```

Arguments

object	object of class Density.Summary

Value

No return value, displays the density summary

`show,Simulation-method`
show

Description

Not currently implemented

Usage

```
## S4 method for signature 'Simulation'
show(object)
```

Arguments

`object`	object of class Simulation

Value

No return value, displays a summary of the simulation

`show,Simulation.Summary-method`
show

Description

Displays the simulation summary

Usage

```
## S4 method for signature 'Simulation.Summary'
show(object)
```

Arguments

`object`	object of class Simulation.Summary

Value

No return value, displays information in Simulation.Summary object

`Simulation-class` *Class "Simulation"*

Description

Class `"Simulation"` is an S4 class containing descriptions of the region, population, survey design and analyses the user wishes to investigate. Once the simulation has been run the N.D.Estimates will contain multiple estimates of abundance and density obtained by repeatedly generating populations, simulating the survey and completing the analyses.

Slots

`reps` Object of class `"numeric"`; the number of times the simulation should be repeated.

`single.transect.set` Object of class `"logical"`; if `TRUE` the same set of transects are used in each repetition.

`design` Object of class `"Survey.Design"`; the survey design.

`population.description` Object of class `"Population.Description"`; the population.description.

`detectability` Object of class `"Detectability"`; a description of the detectability of the population.

`ds.analysis` Object of class `"DS.Analysis"`

`add.options` a list to expand simulation options in the future.

`ddf.param.ests` Object of class `"array"`; stores the parameters associated with the detection function.

`results` A `"list"` with elements 'individuals' (and optionally 'clusters' and 'expected.size') as well as 'Detection'.

The 'individuals' and 'clusters' elements are a list of three 3-dimensional arrays. The first is a summary array containing values for 'Area' (strata area), 'CoveredArea' (the area covered in the strata by the survey), Effort' (the line length or number of points surveyed), 'n' (the number of sightings), 'n.miss.dists' (the number of missing distances - only applicable to mixed detector types and not yet implemented in dsims), 'k' (the number of transects), 'ER' (encounter rate), 'se.ER' (standard error of the encounter rate), 'cv.ER' (coefficient of variation of the encounter rate). A value is provided for each of these for each strata as well as the region as a whole and for each simulation repetition as well as storing the mean and standard deviation of these values across simulation repetitions.

The second array 'N' is the abundance estimates table. It contains values for the 'Estimate' (estimated abundance based on data from iteration i), 'se' (standard error associated with the estimate), 'cv' (coefficient of variation of estimate), 'lcl' (lower 95% confidence interval value), 'ucl' (upper 95% confidence interval value), 'df' the degrees of freedom associated with the estimate. A value is provided for each of these for each strata as well as the region as a whole and for each simulation repetition as well as storing the mean and standard deviation of these values across simulation repetitions.

The third array 'D' is the density estimates table. It contains values for the 'Estimate' (estimated density based on data from iteration i), 'se' (standard error associated with the estimate), 'cv' (coefficient of variation of estimate), 'lcl' (lower 95% confidence interval value),

'ucl' (upper 95% confidence interval value), 'df' the degrees of freedom associated with the estimate. A value is provided for each of these for each strata as well as the region as a whole and for each simulation repetition as well as storing the mean and standard deviation of these values across simulation repetitions.

When animals occur in clusters the expected.size element of the results list contains a 3-dimensional array. It gives values for 'Expected.S' (expected cluster size), 'se.Expected.S' (the standard error of the expected cluster size), 'cv.Expected.S' (the coefficient of variation for the expected cluster size). Values are given for each analysis strata as well as a value for the survey region as a whole and across each simulation repetition as well as overall means and standard deviations across repetitions.

The Detection element of the results list is a 3-dimensional array with values for 'True.Pa' (the proportion of animals in the covered region which were detected), 'Pa' (the estimated proportion of animals detected in the covered region), 'ESW' (the estimated strip width), 'f(0)' (The estimated value of the detection function pdf at distance 0), 'SelectedModel' (the index of the model which had the best fit to the dataset for the repetition), 'DeltaCriteria' (the difference in information criteria between the best and second best fitting models where two or more models were fitted and converged), 'SuccessfulModels' (the number of models which successfully converged). Currently detection functions are pooled across all strata so there is only one global value for each simulated dataset as well as a mean value and standard deviation where appropriate.

`warnings` A `"list"` to store warnings and error messages encountered during runtime.

Methods

`summary signature=(object = "Simulation")`: produces a summary of the simulation and its results.

`generate.population signature = (object = "Simulation")`: generates a single instance of a population.

`generate.transects signature = (object = "Simulation")`: generates a single set of transects.

`run.survey signature = (object = "Simulation")`: carries out the simulation process as far as generating the distance data and returns an object containing the population, transects and data.

`run.simulation signature = (simulation = "Simulation")`: runs the whole simulation for the specified number of repetitions.

See Also

`make.simulation`

`Simulation.Summary-class`

Class "Simulation.Summary"

Description

Class "Simulation.Summary" is an S4 class containing a summary of the simulation results. This is returned when summary(Simulation) is called. If it is not assigned to a variable the object will be displayed via the show method.

Methods

show signature=(object = "Simulation.Summary"): prints the contents of the object in a user friendly format.

---

summary,Density-method

summary

Description

Provides a summary table of the density object.

Usage

```
## S4 method for signature 'Density'
summary(object, ...)
```

Arguments

object	object of class Simulation
...	not implemented

Value

a Density.Summary-class object

---

summary,Simulation-method

summary

Description

Provides a summary of the simulation results.

Usage

```
## S4 method for signature 'Simulation'
summary(object, description.summary = TRUE, use.max.reps = FALSE, ...)
```

Arguments

object	object of class Simulation
description.summary	logical indicating whether an explanation of the summary should be displayed
use.max.reps	by default this is FALSE meaning that only simulation repetitions where all models converged for that data set are included. By setting this to TRUE any repetition where one or more models converged will be included in the summary results.
...	no additional arguments currently implemented

Value

Object of class Simulation.Summary

Survey-class *Virtual Class "Survey"*

Description

Class "Survey" is an S4 class containing an instance of a population.

Slots

population Object of class "Population"; an instance of a population.

Survey.LT-class *Class "Survey.LT" extends class "Survey"*

Description

Class "Survey.LT" is an S4 class containing a population and a set of transects.

Slots

transect Object of class "Line.Transect"; the line transects.

perpendicular.truncation Object of class "numeric"; the maximum distance from the transect at which animals may be detected.

See Also

make.design

`Survey.PT-class` *Class "Survey.PT" extends class "Survey"*

Description

Class `"Survey.PT"` is an S4 class containing a population and a set of transects.

Slots

`transect` Object of class `"Point.Transect"`; the point transects.

`radial.truncation` Object of class `"numeric"`; the maximum distance from the transect at which animals may be detected.

See Also

`make.design`

Index