Package 'dssd'

February 20, 2020

Imports sf, plot3D, rgdal, methods

Suggests knitr, rmarkdown, tibble, testthat

VignetteBuilder knitr

Type Package

Title Distance Sampling Survey Design

Version 0.2.1

Author Laura Marshall <lhm@st-andrews.ac.uk>

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

Description Creates survey designs for distance sampling surveys. These designs can be assessed for various effort and coverage statistics. Once the user is satisfied with the design characteristics they can generate a set of transects to use in their distance sampling survey. Many of the designs implemented in this R package were first made available in our 'Distance' for Windows software and are detailed in Chapter 7 of Advanced Distance Sampling, Buckland et. al. (2008, ISBN-13: 978-0199225873). Find out more about estimating animal/plant abundance with distance sampling at <http://distancesampling.org/>.

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

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Collate 'Class.Constructors.R' 'Coverage.Grid.R' 'Transect.R' 'Region.R' 'generic.functions.R' 'Survey.Design.R' 'Line.Transect.Design.R' 'Line.Transect.R' 'Point.Transect.Design.R' 'Point.Transect.R' 'Segment.Transect.Design.R' 'Segment.Transect.R' 'calc.region.width.R' 'calculate.trackline.pl.R' 'calculate.trackline.segl.R' 'calculate.trackline.zz.R' 'calculate.trackline.zzcom.R' 'check.line.design.R' 'check.point.design.R' 'check.shape.R' 'dssd-package.R'

'generate.eqspace.zigzags.R' 'generate.parallel.lines.R'
'generate.random.points.R' 'generate.segmented.grid.R'
'generate.systematic.points.R' 'get.intersection.points.R'
'line.coords.as.dataframe.R' 'point.coords.as.dataframe.R'
'run.coverage.R' 'write.transects.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2020-02-20 05:40:02 UTC

R topics documented:

dssd-package . . . 3
Coverage.Grid-class . . . 3
generate.transects . . . 4
get.area . . . 5
get.coverage . . . 6
Line.Transect-class . . . 7
Line.Transect.Design-class . . . 7
make.coverage . . . 8
make.design . . . 9
make.region . . . 13
plot,Coverage.Grid,ANY-method . . . 14
plot,Line.Transect,ANY-method . . . 15
plot,Region,ANY-method . . . 15
plot,Survey.Design,ANY-method . . . 16
Point.Transect-class . . . 17
Point.Transect.Design-class . . . 17
Region-class . . . 18
run.coverage . . . 18
Segment.Transect-class . . . 19
Segment.Transect.Design-class . . . 20
show,Line.Transect-method . . . 20
show,Survey.Design-method . . . 21
Survey.Design-class . . . 21
Transect-class . . . 22
write.transects . . . 23

Index **26**

dssd-package *Distance Sampling Survey Design (dssd)*

Description

Creates survey designs for distance sampling surveys. These designs can be assessed for various effort and coverage statistics. Once the user is satisfied with the design characteristics they can generate a set of transects to use in their distance sampling survey. Many of the designs implemented in this R package were first made available in our 'Distance' for Windows software and are detailed in Chapter 7 of Advanced Distance Sampling, Buckland et. al. (2008, ISBN-13: 978-0199225873).

Details

The main functions in this pacakge are: make.region, make.design, generate.transects and run.coverage. See also write.transects for examples of how to export surveys generated by dssd.

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 dssd vignettes 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>

Coverage.Grid-class *Class "Coverage.Grid"*

Description

Class "Coverage.Grid" is an S4 class containing descriptions of a grid used to assess the coverage scores of different designs.

Slots

`grid` sf multipoint object

`spacing` the spacing used to create the coverage grid

Objects from the Class

Objects can be created by calls of the form `make.grid(region = make.region(),no.points = 1000,spacing = numeric(0)`

Methods

`plot signature(x = "Coverage.Grid",y = "missing")`: plots the grid of points.

`generate.transects` *S4 generic method to generate an instance of a design*

Description

Uses the survey design details in the design class to generate a set of transects, i.e. a single survey.

Usage

```
generate.transects(object, quiet = FALSE, ...)

## S4 method for signature 'Line.Transect.Design'
generate.transects(object, quiet = FALSE, ...)

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

Arguments

`object`	an object which inherits from class Survey.Design
`quiet`	if TRUE silences some warnings
`...`	optional arguments used for internal calls

Details

The transects are returned within an object of class Transect which records some of the design options used to generate it along with the samplers as an sf object of class 'POINT' or 'LINESTRING'/'MULTILINESTRING'. The Transect object also contains the covered areas as a 'POLYGON' or 'MULTIPOLYGON' sf object.

Value

an object of class Transect

Author(s)

L Marshall

See Also

write.transects

Examples

```
#Point transect example
shapefile.name <- system.file("extdata", "TrackExample.shp", package = "dssd")
region <- make.region(region.name = "study area",
                      shape = shapefile.name)
design <- make.design(region = region,
                      transect.type = "point",
                      design = "random",
                      samplers = 25,
                      design.angle = 45,
                      edge.protocol = "minus",
                      truncation = 3)
# Generate a single survey instance
survey <- generate.transects(design)
plot(region, survey, covered.area = TRUE)

#Line transect example
# Define the design
design <- make.design(region = region,
                      transect.type = "line",
                      design = c("systematic"),
                      line.length = 1000,
                      design.angle = c(179),
                      edge.protocol = "minus",
                      truncation = 1)

# Create a single set of transects to check
survey <- generate.transects(design)
plot(region, survey, covered.area = TRUE)
```

---

get.area *Returns the area of the region*

Description

Returns the area of the region

Usage

```
get.area(object)

## S4 method for signature 'Region'
get.area(object)
```

Arguments

object	object of class Region

Value

numeric value specifying the area of the region

`get.coverage` *S4 generic method to extract coverage scores*

Description

Obtains the coverage scores from the survey design object.

Usage

```
get.coverage(object, strata.id = "all")

## S4 method for signature 'Survey.Design'
get.coverage(object, strata.id = "all")
```

Arguments

`object`	an object which inherits from class Survey.Design
`strata.id`	either "all" or a numeric value indicating the strata index.

Details

See ?make.design for example code

Value

a vector of coverage scores

See Also

make.design

Line.Transect-class *Class "Line.Transect" extends Class "Transect"*

Description

Class "Line.Transect" is an S4 class detailing a set of transects from a point transect design.

Slots

`line.length` the total line length for the transect set

`trackline` the total on and off effort trackline length from the start of the first transect to the end of the last

`cyclictrackline` the trackline distance plus the distance required to return from the end of the last transect to the beginning of the first

See Also

`make.design`

Line.Transect.Design-class *Class "Line.Transect.Design" extends Class "Survey.Design"*

Description

Class "Line.Transect.Design" is an S4 class detailing the type of line transect design.

Slots

`line.length` Numeric value defining the total line length to be generated (may be multiple values relating to each stratum).

`bounding.shape` relevant for zigzag designs, either a minimum bounding "rectangle" or a "convex hull".

Methods

`generate.transects signature=(object = "Line.Transect.Design", quiet = FALSE, ...)`: generates a set of transects from the design.

See Also

`make.design`

`make.coverage` *Creates a Coverage.Grid object*

Description

This creates an instance of the Coverage.Grid class.

Usage

```
make.coverage(
  region = make.region(),
  spacing = numeric(0),
  n.grid.points = 1000
)
```

Arguments

`region`	the region name
`spacing`	spacing to be used to create the coverage grid. If spacing is specified then any value supplied for n.grid.points will be ignored.
`n.grid.points`	the desired number of grid points (note that the exact number generated may differ slightly depending on the shape of the study region).

Value

object of class Coverage.Grid

Author(s)

Laura Marshall

Examples

```
# This example will take a bit of time to generate
# A coverage grid in a rectangular region of 2000 x 500
region <- make.region()
cover <- make.coverage(region, spacing = 50)
plot(region, cover)
# Create coverage grid by approx number of grid points
cover <- make.coverage(region, n.grid.points = 1000)
plot(region, cover)


# Fast running example for CRAN testing purposes
# This spacing is too sparse to assess coverage in a real example
region <- make.region()
cover <- make.coverage(region, spacing = 250)
plot(region, cover)
```

make.design *Creates a Survey.Design object*

Description

Creates a description of a survey design. Designs may use different types of either point or line transect designs across strata but cannot mix point and line transect design types within a single design object.

Usage

```
make.design(
  region = make.region(),
  transect.type = "line",
  design = "systematic",
  samplers = numeric(0),
  line.length = numeric(0),
  seg.length = numeric(0),
  effort.allocation = numeric(0),
  design.angle = 0,
  spacing = numeric(0),
  edge.protocol = "minus",
  seg.threshold = numeric(0),
  bounding.shape = "rectangle",
  truncation = 1,
  coverage.grid = NULL
)
```

Arguments

region	an object of class Region defining the survey region.
transect.type	character variable specifying either "line" or "point"
design	a character variable describing the type of design. Either "random", "systematic", "eszigzag" (equal-spaced zigzag), "eszigzagcom" (equal spaced zigzag with complementary lines) or "segmentedgrid". See details for more information.
samplers	the number of samplers you wish the design to generate (note that the number actually generated may differ slightly due to the shape of the study region for some designs). This may be one value or a value for each stratum.
line.length	the total line length you desire or a vector of line lengths the same length as the number of strata.
seg.length	the length of the line transect segments for a segmented grid design.
effort.allocation	numeric values used to indicate the proportion of effort to be allocated to each strata from number of samplers or line length. If length is 0 (the default) and

only a total line length or total number of samplers is supplied, effort is allocated based on stratum area.

`design.angle` numeric value detailing the angle of the design. Can provide multiple values relating to strata. The use of the angle varies with design, it can be either the angle of the grid of points, the angle of lines or the design axis for the zigzag design. See details. In addition, a value of -1 will cause a random design angle to be generated.

`spacing` used by systematic designs, numeric value(s) to define spacing between transects. Can be a vector of values with one value per stratum.

`edge.protocol` character value indicating whether a "plus" sampling or "minus" sampling protocol is used. See details.

`seg.threshold` this is a percentage threshold value applicable to segmented grid designs controlling which partial segments are discarded around the survey region boundary. By default, the value of 50, means that only segments that are more than half inside the survey region will be retained. To retain all segments, no matter how small they are when clipped to the survey region boundary set this value to 0.

`bounding.shape` only applicable to zigzag designs. A character value saying whether the zigzag transects should be generated using a minimum bounding "rectangle" or a "convex hull". The default is a minimum bounding rectangle.

`truncation` A single numeric value describing the longest distance at which an object may be observed. Truncation distance is constant across strata.

`coverage.grid` An object of class Coverage.Grid for use when running the coverage simulation.

Details

Plus versus Minus Sampling If you choose for your design to use a minus sampling strategy then transects will only be generated within the survey region and will give lower coverage around the edge of the survey region. Plus sampling generates transects within an area greater than the study region. To do this **dssd** first puts a buffer around the study region before generating the transects within the buffered region. The width of the buffer is the truncation distance supplies by the user. Plus sampling helps to ensure more even coverage around the edge of the study area. See *Buckland et. al, 2001* "Introduction to Distance Sampling" for information on when to use plus versus minus sampling.

Point Transect Designs For point transect designs the user may either specify "random" or "systematic" for the design argument. If the user specifies "random", they should also provide a value for effort detailing the number of point transects they wish their survey to have. For stratified designs they may specify a vector of numbers detailing the number of transects per strata or alternatively use the effort.allocation argument to allocate a total effort amount proportionally. If effort.allocation is left blank then effort will be allocated according to strata area. If the user specified"systematic" they may either provide their desired number of samplers or a value for spacing which defines the gap between each of the points (again a vector of spacing values can be provided for each strata). Optionally the user may select a design.angle. For both random and systematic point transect designs the user may select either a minus or plus sampling edge protocol.

Line Transect Designs: For line transect designs the user may either specify "random" (randomly placed full width lines), "systematic" (systematically placed full width lines), "eszigzag" (equally spaced zigzag lines), "eszigzagcom" (two sets of complementary equally spaced zigzag lines) or

"segmentedgrid" (a grid of short line transect segments). If the user specifies "random", they should provide the either the number of samplers they wish the design to generate or the line length they wish to achieve, either by strata or as a total. If the user specifies "systematic" they should specify either the number of samplers, the desired line length or the spacing between lines. The design angle for these parallel line designs refers to the angle of the lines where 0 is a vertical line and moving round in a clockwise direction. If the user specifies a zigzag design they should specify the systematic spacing value, number of samplers or line length to be used and should choose between generating the design in a minimum bounding rectangle or a convex hull. The default is minimum bounding rectangle which gives more even coverage but the convex hull is generally more efficient. A segmented grid design may be generated using the either the number of samplers or total line length, combined with a value for segment length. Alternatively the user may specify a values for spacing and segment length. The segmented grid design also uses the segment threshold argument. All the designs may be generated using plus or minus sampling protocols. Similar to the point transect designs different values may be specified for each strata for all of the above options. The design angle for the zigzag designs refers to the angle of a line which would run through the middle of each zigzag transect if the zigzags were to be generated within a rectangle. The design angle for zigzags should usually run along the longest dimension of the study region.

See the Getting Started Vignette and the Multiple Strata in dssd Vignette for example designs.

Value

object of a class which inherits from class Survey.Design either Line.Transect.Design or Point.Transect.Design

Author(s)

Laura Marshall

Examples

```
#Point transect example
shapefile.name <- system.file("extdata", "TrackExample.shp", package = "dssd")
region <- make.region(region.name = "study area",
                      shape = shapefile.name)

# Generate coverage grid
cover <- make.coverage(region,
                       n.grid.points = 500)

# Define design
design <- make.design(region = region,
                      transect.type = "point",
                      design = "random",
                      samplers = 25,
                      design.angle = 45,
                      edge.protocol = "minus",
                      truncation = 3,
                      coverage.grid = cover)

# Generate a single survey instance
survey <- generate.transects(design)
plot(region, survey, covered.area = TRUE)
```

```
# Warning! this will take some time to run
design <- run.coverage(design, reps = 500)
# Plot the coverage
plot(design)
# Display the design statistics
design
#Extract coverage scores
coverage.scores <- get.coverage(design)
hist(coverage.scores)

#Multi-strata line transect example
shapefile.name <- system.file("extdata", "AreaRProjStrata.shp", package = "dssd")
region <- make.region(region.name = "study area",
                      strata.name = c("North", "NW", "West Upper",
                                      "West Lower", "SW", "South"),
                      shape = shapefile.name)
plot(region)
# Make a coverage grid
cover <- make.coverage(region,
                       n.grid.points = 500)
# Define the design
design <- make.design(region = region,
                      transect.type = "line",
                      design = c("systematic", "systematic",
                                 "eszigzag", "systematic",
                                 "systematic", "eszigzagcom"),
                      line.length = 5000*1000, #5000km x 1000m (projection in m)
                      design.angle = c(160, 135, 170, 135, 50, 60),
                      edge.protocol = "minus",
                      truncation = 3000,
                      coverage.grid = cover)

# Create a single set of transects to check
survey <- generate.transects(design)
plot(region, survey, covered.area = TRUE)

# Warning! this will quite a long time to run as it is a complex example.
design <- run.coverage(design, reps = 500)
# Plot the coverage
plot(design)
# Display the design statistics
design
#Extract coverage scores for the first strata
coverage.scores <- get.coverage(design, strata.id = 1)
summary(coverage.scores)


# Fast running example for CRAN testing purposes
# This spacing is too sparse to assess coverage in a real example and
# the number of repetitions is too low to assess design statistics
cover <- make.coverage(region,
                       n.grid.points = 50)
```

```
design <- make.design(region = region,
                      transect.type = "point",
                      design = "random",
                      samplers = 25,
                      design.angle = 45,
                      edge.protocol = "minus",
                      truncation = 3,
                      coverage.grid = cover)
survey <- generate.transects(design)
plot(region, survey, covered.area = TRUE)
design <- run.coverage(design, reps = 3)
plot(design)
design
```

`make.region` *Creates a Region object*

Description

This creates an instance of the Region class which defines the study area for the survey.

Usage

```
make.region(
  region.name = "region",
  strata.name = character(0),
  units = character(0),
  shape = NULL
)
```

Arguments

`region.name`	the region name
`strata.name`	the stratum names (character vector, same length as the number of areas in the shapefile / sf object). If not supplied "A", "B", "C", ... will be assigned.
`units`	measurement units; either `"m"` for metres or `"km"` for kilometres. If the shapefile has a projection file associated with it the units will be taken from there.
`shape`	shapefile path to .shp file or an sf object of class sf, sfc or sfg.

Value

object of class Region

Author(s)

Laura Marshall

Examples

```
# A basic study rectangular study region
region <- make.region()
plot(region)

#Load the region from a projected shapefile
shapefile.name <- system.file("extdata", "TrackExample.shp", package = "dssd")
region <- make.region(region.name = "study area",
                      shape = shapefile.name)
plot(region)

#Load a multi strata unprojected shapefile
shapefile.name <- system.file("extdata", "AreaRStrata.shp", package = "dssd")
# Need to load shapefile first as it is not projected
sf.shape <- sf::read_sf(shapefile.name)
# Check current coordinate reference system
sf::st_crs(sf.shape)
# Define a European Albers Equal Area projection
proj4string <- "+proj=aea +lat_1=43 +lat_2=62 +lat_0=30 +lon_0=-9 +x_0=0 +
                y_0=0 +ellps=intl +units=km"
# Project the study area on to a flat plane
projected.shape <- sf::st_transform(sf.shape, crs = proj4string)
# Create region with default strata names
region <- make.region(region.name = "study area",
                      shape = projected.shape)
# By plotting the region we can verify the order of the strata
plot(region)
```

---

plot,Coverage.Grid,ANY-method

Plot

Description

Plots an S4 object of class 'Coverage.Grid'

Usage

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

Arguments

x	object of class Coverage.Grid
y	not used
...	other general plot parameters including: add as TRUE / FALSE, col as a colour for grid points and pch as grid point symbols.

plot,Line.Transect,ANY-method
Plot

Description

Plots an S4 object of class 'Transect'

Usage

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

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

Arguments

x	object of class transect
y	not used
...	Additional arguments: add (TRUE/FALSE) whether to add to existing plot, col colour, lwd line width (for line transects) and pch point symbols (for point transects).

plot,Region,ANY-method
Plot

Description

Plots an S4 object of class 'Region'

Usage

```
## S4 method for signature 'Region,ANY'
plot(
  x,
  y,
  main = "",
  region.col = "default",
  legend.params = list(inset = c(-0.2, 0), cex = 0.75, wrap = 15),
  ...
)

## S4 method for signature 'Region,Transect'
```

```
plot(
  x,
  y,
  main = "",
  region.col = "default",
  subtitle = "",
  covered.area = FALSE,
  ...
)

## S4 method for signature 'Region,Coverage.Grid'
plot(x, y, main = "", region.col = "default", subtitle = "", ...)
```

Arguments

x	object of class Region or inheriting from Survey
y	optionally a Survey object to plot with the Region
main	the main title for the plot
region.col	colours for the strata
legend.params	a list of parameters which affect the location and appearance of the legend. 'inset' affects the location of the legend, 'cex' affects the text size and 'wrap' is the number of character in a line before the text is wrapped on to the next line.
...	Additional plot arguments passed to the plot method for the y argument.
subtitle	a subtitle for the plot
covered.area	boolean value saying whether the covered area should be plotted.

---

plot,Survey.Design,ANY-method *Plot*

Description

Plots the coverage scores contained within an object of class 'Survey.Design' and provides a colour key relating to the coverage scores. This allows the user to assess how even the coverage is across the survey region.

Usage

```
## S4 method for signature 'Survey.Design,ANY'
plot(x, y, strata.id = numeric(0), col.breaks = 10, subtitle = "", ...)
```

Arguments

x	object of class Survey.Design
y	not used
strata.id	a numeric value indicating the index of the strata you wish to plot.
col.breaks	the number of break point in the colour scale representing the coverage scores.
subtitle	a subtitle for the plot.
...	not implemented for this class.

Point.Transect-class *Class "Point.Transect" extends Class "Survey"*

Description

Virtual Class "Point.Transect" is an S4 class detailing a set of transects from a point transect design.

See Also

make.design

Point.Transect.Design-class *Virtual Class "Point.Transect.Design" extends Class "Survey.Design"*

Description

Virtual Class "Point.Transect.Design" is an S4 class detailing the type of point transect design.

Methods

generate.transects signature=(object = "Point.Transect.Design",quiet = FALSE,...): generates a set of transects from the design.

See Also

make.design

Region-class *Class "Region"*

Description

Class "Region" is an S4 class containing descriptions of the study area. Uses an object of class

Slots

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

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

units Object of class "character"; character describing the coordinate units ("km" or "m")

area Object of class "numeric"; the area of the survey region

region Object of class "sf" defining the survey region

Objects from the Class

Objects can be created by calls of the form make.region(region.name = "region.name",shapefile = region.shapefile)

Methods

get.area signature(obj = "Region"): retrieves the area element

plot signature(x = "Region",y = "missing"): plots the survey region defined by the object.

See Also

make.region

run.coverage *run.coverage*

Description

This function can be used to assess the coverage of a design and also assess design statistics, such as how the number of samplers, the line length, trackline length or percentage coverage varies between surveys generated from the same design. It generates the specified number of surveys from the design and looks to see which of the coverage grid points, a systematic grid of points across the survey region, are included in each survey. When calculating coverage scores if more than one sampler falls on a grid point then that grid point gets allocated the appropriate count. These counts are then averaged over the number of surveys which have been generated. At the same time it records the relevant statistics for the design. While 100 repetitions may be sufficient to get an idea of design statistics 1000 or even more repetitions may be needed to gain a good representation of the coverage scores across the study region.

Usage

```
run.coverage(design, reps = 10, save.transects = "", quiet = FALSE)
```

Arguments

design	an object which inherits from the Survey.Design class.
reps	the number of times you wish the coverage simulation to be carried out.
save.transects	a directory where the shapefiles for the transects can be saved. The shapefile names will be S1, S2, ... existing files in the directory will not be overwritten.
quiet	when TRUE no progress counter is displayed.

Details

See ?make.design for example code.

Value

this function returns the survey design object passed in and it will now include the coverage and design statistics.

See Also

make.design

Segment.Transect-class

Class "Segmemt.Transect" extends Class "Line.Transect"

Description

Class "Segment.Transect" is an S4 class detailing a set of transects from a point transect design.

Slots

seg.length length of the transect segment.

seg.threshold this is a percentage threshold value applicable to segmented grid designs controlling which partial segments are discarded around the survey region boundary. By default, the value of 50, means that only segments that are more than half inside the survey region will be retained. To retain all segments, no matter how small they are when clipped to the survey region boundary set this value to 0.

offset a value to offset a return transect by so segments become pairs of segments (not yet implemented).

See Also

make.design

Segment.Transect.Design-class

Class "Segment.Transect.Design" extends Class "Survey.Design"

Description

Class "Segment.Transect.Design" is an S4 class detailing the a segmented line transect design.

Slots

`seg.length` length of the transect segment.

`seg.threshold` this is a percentage threshold value applicable to segmented grid designs controlling which partial segments are discarded around the survey region boundary. By default, the value of 50, means that only segments that are more than half inside the survey region will be retained. To retain all segments, no matter how small they are when clipped to the survey region boundary set this value to 0.

`offset` a value to offset a return transect by so segments become pairs of segments (not yet implemented).

Methods

`generate.transects signature=(object = "Line.Transect.Design",quiet = FALSE,...)`: generates a set of transects from the design.

See Also

`make.design`

show,Line.Transect-method

Show

Description

Displays details of an S4 object of class 'Transect'

Displays details of an S4 object of class 'Transect'

Usage

```
## S4 method for signature 'Line.Transect'
show(object)

## S4 method for signature 'Point.Transect'
show(object)
```

Arguments

object	an object of class Transect

---

show,Survey.Design-method *show*

Description

Summarises and displays an S4 object of class 'Survey.Design'

Usage

```
## S4 method for signature 'Survey.Design'
show(object)
```

Arguments

object	an object which inherits from the Survey.Design class

Survey.Design-class *Virtual Class "Survey.Design"*

Description

Virtual Class "Survey.Design" is an S4 class detailing the survey design.

Slots

region An object of class 'Region' defining the study area.

design Character value describing the name of the design.

samplers Numeric values defining the number of samplers in each stratum.

effort.allocation numeric values used to indicate the proportion of effort to be allocated to each strata from number of samplers or line length. If length 0, effort allocated based on stratum area.

spacing used by systematic designs, numeric value to define spacing between transects.

design.angle numeric value detailing the angle of the design. Can provide multiple values relating to strata. The use of the angle varies with design, it can be either the angle of the grid of points, the angle of lines or the design axis for the zigzag design.

edge.protocol Character value defining whether a "minus" or "plus" sampling strategy should be used.

truncation Object of class "numeric"; The maximum distance at which observations can be made. This is used to determine the covered area during the coverage calculations.

`coverage.grid` The coverage grid used to assess the uniformity of coverage during simulations.

`coverage.scores` The average number of times each point in the coverage grid is included in a survey.

`coverage.reps` The number of times the coverage simulation was repeated.

`design.statistics` A list of values obtained when investigating coverage. This includes the minimum, maximum, mean and median

Methods

`generate.transects signature 'Survey.Design'`: Generates a set of transects from the design.

`plot signature 'Survey.Design,ANY'`: Plots the coverage scores contained within an object of class 'Survey.Design' and provides a colour key relating to the coverage scores. This allows the user to assess how even the coverage is across the survey region.

`show signature 'Survey.Design'`: Gives a summary of the design description, stratum areas and coverage scores if the coverage simulation has been run on the design. The coverage score summary details the minimum, maximum, mean and medium coverage scores across the study region. Also gives summaries of other design measures such as the number of samplers, line length, trackline length, cyclic trackline length, covered area and percentage of region covered.

See Also

`make.design`

Transect-class *S4 Class "Transect"*

Description

Virtual Class "Transect"

Details

Virtual Class `"Transect"` is an S4 class detailing a single survey, a single set of transects.

Slots

`strata.names` a character vector of the strata names

`design` Describes the design algorithm used to create the survey.

`samplers` Contains the survey transects

`strata.area` The areas of the strata in the design

`cov.area` The total areas sampled within each strata. Areas sampled twice are counted twice.

`cov.area.polys` The polygons representing the covered area of the survey.

`samp.count` Numeric value(s) giving the number of realised transects.

`effort.allocation` a vector of probabilities determining how effort is allocated between strata. Effort allocated based on area if left empty.

`spacing` determines the spacing of systematic samplers

`design.angle` numeric value detailing the angle of the design. Can provide multiple values relating to strata. The use of the angle varies with design, it can be either the angle of the grid of points, the angle of lines or the design axis for the zigzag design.

`edge.protocol` character value indicating whether a "plus" sampling or "minus" sampling protocol is used.

See Also

`make.design`

`write.transects` *Writes transects to file*

Description

This function will write a set of transects to file, either as a shapefile or gpx file, or it will write the transect coordinates (centre points for point transects or end points for line transects) to a comma-separated values 'csv' file or a text file 'txt' with tabular spacing between columns. For line transects which have been split across geographical features (such as islands or lakes) there will be two or more rows in the csv / txt file with all rows having the same transect ID.

Usage

```
write.transects(
  object,
  dsn,
  layer = character(0),
  dataset.options = character(0),
  overwrite = FALSE,
  proj4string = character(0)
)
```

Arguments

`object`	an object inheriting from class Transect or an sf spatial object extracted from a Transect object.
`dsn`	the data source name, currently a filename with a 'shp' 'csv', 'txt' or 'gpx' extension.
`layer`	a character vector specifying the layer name, only required for gpx files.
`dataset.options`	a character vector of options, which vary by driver, and should be treated as experimental. Used to specify "GPX_USE_EXTENSIONS=yes" for writing gpx files.

overwrite	whether or not existing files should be overwritten. Only applicable when writing to gpx files.
proj4string	The projection you wish the coordinates of the output file to be in. Note, when writing to gpx file the transect coordinates must be in latitude and longitude.

Details

To write the transects to shapefile only the dsn is needed with a 'shp', 'csv' or 'txt' file extension. To write a gpx file you need to specify the dsn, layer, dataset.options and usually a projection to project the coordinates back into latitude and longitude.

Value

invisibly the Transect object

Author(s)

Laura Marshall

Examples

```
# Note that for CRAN testing purposes all files written in example code must
# be written to a temporary directory, to view this location type tempdir().
# It is however advised that you replace the tempdir() commands in the code
# below to a more easily accessible directory to which the files will be
# written.

# Make the default design in the default study area
design <- make.design()
transects <- generate.transects(design)
write.transects(transects, dsn = paste0(tempdir(), "/", "transects.shp"))

# Writing csv file example
write.transects(transects, dsn = paste0(tempdir(), "/", "transects.csv"))

# Writing txt file example
write.transects(transects, dsn = paste0(tempdir(), "/", "transects.txt"))

# Writing gpx file example - must project transect coords into lat/lon
#Load the unprojected shapefile
shapefile.name <- system.file("extdata", "TentsmuirUnproj.shp", package = "dssd")
sf.shape <- sf::read_sf(shapefile.name)
# Check current coordinate reference system
orig.crs <- sf::st_crs(sf.shape)
# Define a European Albers Equal Area projection
proj4string <- "+proj=aea +lat_1=56 +lat_2=62 +lat_0=50 +lon_0=-3 +x_0=0
                +y_0=0 +ellps=intl +units=m"
# Project the study area on to a flat plane
projected.shape <- sf::st_transform(sf.shape, crs = proj4string)
# Create the survey region in dssd
region.tm <- make.region(region.name = "Tentsmuir",
                         strata.name = c("Main Area", "Morton Lochs"),
```

```
                          shape = projected.shape)

design <- make.design(region = region.tm,
                      transect.type = "line",
                      design = "systematic",
                      samplers = 20,
                      design.angle = 90)
survey <- generate.transects(design)
plot(region.tm, survey)

write.transects(survey,
                dsn = paste0(tempdir(), "/", "transects.gpx"),
                layer = "lines",
                dataset.options = "GPX_USE_EXTENSIONS=yes",
                proj4string = orig.crs)
```

Index

∗Topic **classes**
Coverage.Grid-class, 3
Line.Transect-class, 7
Line.Transect.Design-class, 7
Point.Transect-class, 17
Point.Transect.Design-class, 17
Region-class, 18
Segment.Transect-class, 19
Segment.Transect.Design-class, 20
Survey.Design-class, 21
Transect-class, 22
∗Topic **package**
dssd-package, 3

Coverage.Grid-class, 3

dssd *(*dssd-package*)*, 3
dssd-package, 3

generate.transects, *3*, 4
generate.transects,Line.Transect.Design-method *(*generate.transects*)*, 4
generate.transects,Point.Transect.Design-method *(*generate.transects*)*, 4
get.area, 5
get.area,Region-method *(*get.area*)*, 5
get.coverage, 6
get.coverage,Survey.Design-method *(*get.coverage*)*, 6

Line.Transect-class, 7
Line.Transect.Design-class, 7

make.coverage, 8
make.design, *3*, *6, 7*, 9, *17*, *19, 20*, *22, 23*
make.region, *3*, 13, *18*

plot,Coverage.Grid,ANY-method, 14
plot,Line.Transect,ANY-method, 15
plot,Point.Transect,ANY-method *(*plot,Line.Transect,ANY-method*)*, 15
plot,Region,ANY-method, 15
plot,Region,Coverage.Grid-method *(*plot,Region,ANY-method*)*, 15
plot,Region,Transect-method *(*plot,Region,ANY-method*)*, 15
plot,Survey.Design,ANY-method, 16
Point.Transect-class, 17
Point.Transect.Design-class, 17

Region-class, 18
run.coverage, *3*, 18

Segment.Transect-class, 19
Segment.Transect.Design-class, 20
show,Line.Transect-method, 20
show,Point.Transect-method *(*show,Line.Transect-method*)*, 20
show,Survey.Design-method, 21
Survey.Design-class, 21

Transect-class, 22

write.transects, *3*, *4*, 23