Package: ssarp (via r-universe)

June 1, 2026

Type Package

Title Species-/Speciation-Area Relationship Projector

Description Create Species- and Speciation-Area Relationships using occurrence records or presence-absence matrices.

Maintainer Kristen Martinet <kmartinet@outlook.com>

Version 0.5.1

URL https://github.com/ropensci/ssarp, https://docs.ropensci.org/ssarp

BugReports https://github.com/ropensci/ssarp/issues

Imports ape, checkmate, cli, Dict, dplyr, httr, mapdata, maps, reshape2, segmented, sf, stringi, terra, tidyr

LazyData true

Encoding UTF-8

License GPL (>=2)

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Depends R (>= 4.3.0)

Suggests testthat (>= 3.0.0), rmarkdown, knitr (>= 1.46), rgbif, BAMMtools, curl, httptest, spdep, epm

Config/testthat/edition 3

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libicu-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-06-01 21:51:46 UTC

RemoteUrl https://github.com/ropensci/ssarp

RemoteRef main

RemoteSha 4f0111a31fa9d8dd68c244afd8849bcc6db36f3b

Contents

`create_sar` *Create a species-area relationship (SAR) plot*

Description

Use segmented regression to create a species-area relationship (SAR) plot. The X axis represents log(island area) and the Y axis represents log(number of species)

Usage

```
create_sar(occurrences, npsi = 1, visualize = FALSE)

create_SAR(occurrences, npsi = 1, visualize = FALSE)
```

Arguments

`occurrences`	The dataframe output by `ssarp::find_areas()` (or if using a custom dataframe, ensure that it has the following columns: specificEpithet, areas)
`npsi`	The maximum number of breakpoints to estimate for model selection. If 0 is input, only a linear model will be run. Default: 1
`visualize`	(boolean) Whether the plot should be displayed when the function is called. Default: FALSE

Details

If the user would prefer to create their own plot of the `ssarp::create_sar()` output, the `aggDF` element of the returned list includes the raw points from the plot created here. They can be accessed as demonstrated in the Examples section.

Value

A list of class SAR with 5 items including:

- `summary`: the summary output
- `segObj` or `linObj`: the regression object (`segObj` when segmented, `linObj` when linear)
- `aggDF`: the aggregated dataframe used to create the plot
- `AICscores`: the AIC scores generated during model selection
- `AllModels`: a list of models created in model selection, labeled by number of breakpoints

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
land <- find_land(occurrences = dat)
areas <- find_areas(occs = land)

seg <- create_sar(occurrences = areas,
                  npsi = 1,
                  visualize = FALSE)
plot(seg)
summary <- seg$summary
model_object <- seg$segObj
points <- seg$aggDF
```

`create_spar` *Create a speciation-area relationship plot (SpAR)*

Description

Use segmented regression to create a speciation-area relationship plot. The X axis represents log(island area) and the Y axis represents log(speciation rate)

Usage

```
create_spar(occurrences, npsi = 1, visualize = FALSE)

create_SpAR(occurrences, npsi = 1, visualize = FALSE)
```

Arguments

occurrences	The dataframe output by one of ssarp's speciation methods (ssarp::estimate_bamm(), ssarp::estimate_dr(), ssarp::estimate_ms()), or if using a custom dataframe, ensure that it has the following columns: areas, rate
npsi	The maximum number of breakpoints to estimate for model selection. If 0 is input, only a linear model will be run. Default: 1
visualize	(boolean) Whether the plot should be displayed when the function is called. Default: FALSE

Details

If the user would prefer to create their own plot of the ssarp::create_spar() output, the aggDF element of the returned list includes the raw points from the plot created here. They can be accessed as demonstrated in the Examples section.

More information about the three methods for estimating speciation rate included in ssarp can be found in ssarp's SpAR vignette.

Value

A list of class SAR with 5 items including:

- summary: the summary output
- segObj or linObj: the regression object (segObj when segmented, linObj when linear)
- aggDF: the aggregated dataframe used to create the plot
- AICscores: the AIC scores generated during model selection
- AllModels: a list of models created in model selection, labeled by number of breakpoints

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
land <- find_land(occurrences = dat)
areas <- find_areas(occs = land)

# Read tree from Patton et al. (2021), trimmed to Caribbean species
tree <- ape::read.tree(system.file("extdata",
                                   "Patton_Anolis_Trimmed.tree",
                                   package = "ssarp"))

occ_speciation <- estimate_ms(tree = tree,
                              label_type = "epithet",
                              occurrences = areas)
```

```
seg <- create_spar(occurrences = occ_speciation,
                   npsi = 1,
                   visualize = FALSE)
plot(seg)
summary <- seg$summary
model_object <- seg$segObj
points <- seg$aggDF
```

`estimate_bamm` *Get tip speciation rates from BAMM (Rabosky 2014) analysis*

Description

Use the BAMMtools package (Rabosky et al. 2014) to extract tip speciation rates from user-supplied BAMM analysis objects.

Usage

```
estimate_bamm(label_type = "binomial", occurrences, edata)

estimate_BAMM(label_type = "binomial", occurrences, edata)
```

Arguments

`label_type`	Either "epithet" or "binomial" (default): describes the type of tip label in the tree used for the BAMM analysis. If "epithet," only the species epithet will be used to match speciation rates to tips in the returned occurrence dataframe. If "binomial," the full species name (including genus) will be used to match speciation rates to tips in the returned occurrence dataframe.
`occurrences`	The occurrence record dataframe output from the ssarp pipeline. If you would like to use a custom dataframe, please make sure that there are columns titled "genericName" and "specificEpithet"
`edata`	The eventdata object created by using the `BAMMtools::getEventData()` function

Value

A dataframe that includes speciation rates for each species in the occurrence record dataframe

References

- Rabosky, D.L. (2014). Automatic Detection of Key Innovations, Rate Shifts, and Diversity-Dependence on Phylogenetic Trees. PLOS ONE, 9(2): e89543.
- Rabosky, D.L., Grundler, M., Anderson, C., Title, P., Shi, J.J., Brown, J.W., Huang, H., & Larson, J.G. (2014), BAMMtools: an R package for the analysis of evolutionary dynamics on phylogenetic trees. Methods in Ecology and Evolution, 5: 701-707.

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                            "ssarp_Example_Dat.csv",
                            package = "ssarp"))
land <- find_land(occurrences = dat)
areas <- find_areas(occs = land)

# Read tree from Patton et al. (2021), trimmed to Caribbean species
tree <- ape::read.tree(system.file("extdata",
                             "Patton_Anolis_Trimmed.tree",
                             package = "ssarp"))

# Event data file from an external BAMM run
event_data <- system.file("extdata",
                          "event_data_Patton_Anolis.txt",
                           package = "ssarp")

edata <- BAMMtools::getEventData(phy = tree, eventdata = event_data)

occ_speciation <- estimate_bamm(label_type = "epithet",
                                occurrences = areas ,
                                edata = edata)
```

`estimate_dr` *Get speciation rates using the DR statistic (Jetz et al. 2012)*

Description

DR stands for "diversification rate," but it is ultimately a better estimation of speciation rate than net diversification (Belmaker and Jetz 2015; Quintero and Jetz 2018) and returns results similar to BAMM's tip speciation rate estimations (Title and Rabosky 2019).

Usage

```
estimate_dr(tree, label_type = "binomial", occurrences)

estimate_DR(tree, label_type = "binomial", occurrences)
```

Arguments

`tree`	The dated phylogenetic tree that corresponds with the taxa to be included in a speciation-area relationship

`label_type` Either "epithet" or "binomial" (default): describes the type of tip label in the provided tree. If "epithet," only the species epithet will be used to match speciation rates to tips in the returned occurrence dataframe. If "binomial," the full species name (including genus) will be used to match speciation rates to tips in the returned occurrence dataframe.

`occurrences` The occurrence record dataframe output from the ssarp pipeline. If you would like to use a custom dataframe, please make sure that there are columns titled "genericName" and "specificEpithet"

Details

This function uses methodology from Sun and Folk (2020) to calculate the DR statistic for each tip of a given tree and output a dataframe for use in ssarp's speciation-area relationship pipeline. This method also removes any species rows without rates (this is most likely to occur when the tree does not have all of the species included in the occurrence record dataframe). The tree read in the examples below is modified from Patton et al. (2021).

Value

A dataframe that includes speciation rates for each species in the occurrence record dataframe

References

- Belmaker, J., & Jetz, W. (2015). Relative roles of ecological and energetic constraints, diversification rates and region history on global species richness gradients. Ecology Letters, 18: 563–571.
- Jetz, W., Thomas, G.H, Joy, J.B., Harmann, K., & Mooers, A.O. (2012). The global diversity of birds in space and time. Nature, 491: 444-448.
- Patton, A.H., Harmon, L.J., del Rosario Castañeda, M., Frank, H.K., Donihue, C.M., Herrel, A., & Losos, J.B. (2021). When adaptive radiations collide: Different evolutionary trajectories between and within island and mainland lizard clades. PNAS, 118(42): e2024451118.
- Quintero, I., & Jetz, W. (2018). Global elevational diversity and diversification of birds. Nature, 555, 246–250.
- Sun, M. & Folk, R.A. (2020). Cactusolo/rosid_NCOMMS-19-37964-T: Code and data for rosid_NCOMMS-19-37964 (Version V.1.0). Zenodo. http://doi.org/10.5281/zenodo.3843441
- Title P.O. & Rabosky D.L. (2019). Tip rates, phylogenies and diversification: What are we estimating, and how good are the estimates? Methods in Ecology and Evolution. 10: 821–834.

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
```

```
land <- find_land(occurrences = dat)
areas <- find_areas(occs = land)

# Read tree from Patton et al. (2021), trimmed to Caribbean species
tree <- ape::read.tree(system.file("extdata",
                                   "Patton_Anolis_Trimmed.tree",
                                   package = "ssarp"))

occ_speciation <- estimate_dr(tree = tree,
                              label_type = "epithet",
                              occurrences = areas)
```

`estimate_ms` *Get speciation rates following equation 4 in Magallón and Sanderson (2001)*

Description

Use methodology from Magallón and Sanderson (2001) to estimate speciation rates using a user-provided phylogeny and output a dataframe for use in ssarp's speciation-area relationship pipeline. This method also removes any species rows without rates (this is most likely to occur when the tree does not have all of the species included in the occurrence record dataframe)

Usage

```
estimate_ms(tree, label_type = "binomial", occurrences)

estimate_MS(tree, label_type = "binomial", occurrences)
```

Arguments

`tree`	The dated phylogenetic tree that corresponds with the taxa to be included in a speciation-area relationship
`label_type`	Either "epithet" or "binomial" (default): describes the type of tip label in the provided tree. If "epithet," only the species epithet will be used when interacting with the tree. If "binomial," the full species name (including genus) will be used when interacting with the tree.
`occurrences`	The occurrence record dataframe output from the ssarp pipeline. If you would like to use a custom dataframe, please make sure that there are columns titled "specificEpithet", "genericName", and "areas"

Value

A dataframe that includes speciation rates for each island in the user-provided occurrence record dataframe.

References

- Magallón, S. & Sanderson, M.J. (2001). Absolute Diversification Rates in Angiosperm Clades. Evolution, 55(9): 1762-1780.

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
land <- find_land(occurrences = dat)
areas <- find_areas(occs = land)

# Read tree from Patton et al. (2021), trimmed to Caribbean species
tree <- ape::read.tree(system.file("extdata",
                                   "Patton_Anolis_Trimmed.tree",
                                   package = "ssarp"))

occ_speciation <- estimate_ms(tree = tree,
                              label_type = "epithet",
                              occurrences = areas)
```

`find_areas` *Find areas of land masses.*

Description

Find the areas of the land masses relevant to the taxon of interest with two options: a database of island names and areas, or a user-provided shapefile.

Usage

```
find_areas(occs, area_custom = NULL, shapefile = NULL, names = NULL)
```

Arguments

`occs`	The dataframe that is returned by `ssarp::find_land()`. If using a custom occurrence record dataframe, ensure that it has the following columns: "genericName", "specificEpithet", "decimalLongitude", "decimalLatitude", "first", "second", "third", "datasetKey". The "datasetKey" column is important for GBIF records and identifies the dataset to which the occurrence record belongs. Custom dataframes without this style of data organization should fill the column with placeholder values.

area_custom — A dataframe including names of land masses and their associated areas. This dataframe should be provided when the user would like to bypass using the built-in database of island names and areas. Please ensure that the custom dataframe includes the land mass's area in a column called "AREA" and the name in a column called "Name". (Optional)

shapefile — A shapefile (.shp) containing spatial information for the geographic locations of interest. (Optional)

names — If the user would like to restrict which polygons in the shapefile are included in the returned occurrence record dataframe, they can be specified here as a vector. If the user does not provide a vector, all of the non-NA names in the shapefile will be included (as found in shapefile$name). (Optional)

Details

The first method is to reference a built-in dataset of island names and areas to find the areas of the landmasses relevant to the taxon of interest. The user may also decide to input their own custom dataframe including names of relevant land masses and their associated areas to bypass using *ssarp*'s built-in dataset.

The second method is to reference a user-supplied shapefile containing spatial information for the landmasses of interest in order to determine their areas.

While the word "landmasses" was used heavily in this documentation, users supplying their own custom area dataframe or shapefile are encouraged to use this function in the *ssarp* workflow to create species- and speciation- area relationships for island-like systems such as lakes, fragmented habitat, and mountain peaks.

Value

A dataframe of the species name, island name, and island area

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
occs <- find_land(occurrences = dat)
areas <- find_areas(occs = occs)
```

find_land *Find the name of the land on which the occurrence points were found*

Description

Use various mapping tools to attempt to find the names of land masses where the occurrence points were found.

Usage

```
find_land(occurrences, fillgaps = FALSE)
```

Arguments

occurrences	A dataframe output by `rgbif::occ_search()` or `rgbif::occ_download()` (or if using a custom dataframe, ensure that it has the following columns: decimalLongitude, decimalLatitude, acceptedScientificName, genericName, specificEpithet, datasetKey). The "datasetKey" column is important for GBIF records and identifies the dataset to which the occurrence record belongs. Custom dataframes without this style of data organization should fill the column with placeholder values.
fillgaps	(logical) Attempt to use Photon API to fill in gaps left by `mapdata::map.where()` (TRUE) or only `mapdata::map.where()` results (FALSE, default). While it is powerful, the Photon API does not have a standard location for island names in its returned information, so using it will likely require the returned dataframe to be cleaned by the user.

Value

A dataframe of the species name, longitude, latitude, and three parts of occurrence information. "first" is the name used to describe the largest possible area of land where the occurrence point is found. "second" is the name used to describe the second-largest possible area of land that corresponds with the occurrence point. "third" is the most specific area of land that corresponds with the occurrence point. Functions later in the ssarp pipeline default to checking whether "third" has an entry, then look at "second," and then "first."

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
occs <- find_land(occurrences = dat, fillgaps = FALSE)
```

`find_pam_areas` *Find areas of islands using a presence-absence matrix (PAM)*

Description

Use a presence-absense matrix (PAM) to create a dataframe that can be used to generate a species-area relationship (SAR) or speciation-area relationship (SpAR) in the `ssarp` pipeline.

Usage

```
find_pam_areas(pam, area_custom = NULL)
```

Arguments

`pam`	A presence-absence matrix (PAM), saved as a dataframe. Please ensure that the PAM has species names (include both generic name and specific epithet, with an underscore separating them) as the column names, with the exception of the first column that designates locations, which must be named "Island".
`area_custom`	A dataframe including names of land masses and their associated areas. This dataframe should be provided when the user would like to bypass using the built-in database of island names and areas. Please ensure that the custom dataframe includes the land mass's area in a column called "AREA" and the name in a column called "Name". (Optional)

Details

PAMs summarize the occurrence of species across different geographic locations. The column names of a PAM are species names, with the exception of the first column, which specifies the name of locations. For each cell corresponding to a species/location pair, either a 1 (presence) or a 0 (absence) is input depending on whether the species can be found at that location or not.

Using a PAM, this function will find the areas of the land masses relevant to the taxon of interest with two options: a built-in database of island names and areas, or a user-provided list of island names and areas.

The default method is to reference a built-in dataset of island names and areas to find the areas of the landmasses relevant to the taxon of interest. The user may also decide to input their own custom dataframe including names of relevant land masses and their associated areas to bypass using `ssarp`'s built-in dataset.

While the word "landmasses" was used heavily in this documentation, users supplying their own custom area dataframe or shapefile are encouraged to use this function in the `ssarp` workflow to create species- and speciation- area relationships for island-like systems such as lakes, fragmented habitat, and mountain peaks.

Value

A dataframe of species names, island names, and island areas

Examples

```
pam <- read.csv(system.file("extdata",
                            "example_pam.csv",
                            package = "ssarp"))
areas <- find_pam_areas(pam = pam)
```

`get_island_areas` *Access properly encoded island_area object*

Description

In order to address an R CMD check warning about non-ASCII characters in the island_area object, these characters in the island names had to be converted to an ASCII format. The non-ASCII accents in the island names are important for the functionality of the ssarp package, so this function provides the user with a dataframe including the original, un-converted island names.

Usage

```
get_island_areas()
```

Value

An edited version of the `ssarp::island_areas` object, which is a dataframe including the names, areas, and maximum elevations of islands from across the globe.

Examples

```
island_df <- get_island_areas()
```

`get_presence_absence` *Create a presence-absence dataframe for a given occurrence record dataframe*

Description

Use a dataframe output by `ssarp::find_areas()` to determine which species occur on specific islands by creating a presence-absence dataframe. A 1 represents presence and a 0 represents absence.

Usage

```
get_presence_absence(occs)
```

Arguments

occs — The dataframe output by `ssarp::find_areas()`, or if using a custom dataframe, ensure that it has the following named columns:

- "areas" containing the areas associated with the land masses of interest
- "specificEpithet" containing the names of the species living on those islands
- "first" containing locality information. In the ssarp workflow, this column contains the country name
- "second" containing locality information. In the ssarp workflow, this column contains a province or island name
- "third" containing locality information. In the ssarp workflow, this column contains the island name if the 7th column does not contain the island name

Value

A dataframe with a row for each island in the given occurrence record dataframe and a column for each species. Within each species column, a 1 represents the presence of that species on the island corresponding to the given row, and a 0 represents the absence of that species on the island corresponding to the given row.

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
land <- find_land(occurrences = dat)
areas <- find_areas(occs = land)
pres_abs <- get_presence_absence(areas)
```

---

get_richness — *Create a species richness dataframe for a given occurrence record dataframe*

Description

Use a dataframe output by `ssarp::find_areas()` to determine how many species occur on each island by creating a species richness dataframe.

Usage

```
get_richness(occs)
```

Arguments

occs	The dataframe output by `ssarp::find_areas()`, or if using a custom dataframe, ensure that it has the following named columns: • "areas" containing the areas associated with the land masses of interest • "specificEpithet" containing the names of the species living on those islands

Details

The output of this function can be used directly with the sars R package to fit additional SAR models that `ssarp` does not create itself.

Value

A dataframe with two columns: the first containing island areas and the second containing the associated species richness (number of unique species)

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
land <- find_land(occurrences = dat)
areas <- find_areas(occs = land)
richness <- get_richness(occs = areas)
```

---

get_sources *Gather sources from GBIF data for citation*

Description

When using data obtained via `rgbif::occ_search()` and filtered with `ssarp::find_areas()` for a publication, you must keep a record of the datasets used in your analysis. This function assists in creating the dataframe necessary to follow GBIF's citation guidelines (see References).

Usage

```
get_sources(occs)
```

Arguments

occs	The occurrence record dataframe returned by `rgbif::occ_search()` or `ssarp::find_areas()`.

Value

A dataframe of dataset keys and the number of occurrence records associated with each key that were gathered with `rgbif::occ_search()` and/or filtered with `ssarp::find_areas()`.

References

- GBIF citation guidelines
- Data obtained via `rgbif::occ_search()` and filtered with `ssarp::find_areas()` falls under the derived datasets distinction
- More information about creating derived datasets

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
source_df <- get_sources(occs = dat)
```

`island_areas` *Island name data*

Description

A collection of island names, their areas, and their elevations found using ArcGIS

Usage

```
island_areas
```

Format

'island_areas':

A data frame with 26376 rows and 5 variables:

`OBJECTID_1` integer Object ID from ArcGIS

`COUNT` integer Number of cells within island, from ArcGIS

`AREA` double Area of the island

`MAX` integer Maximum elevation of the island

`Name` character Name of the island

plot.SAR *Plot a species-area relationship*

Description

Function for plotting species-area relationship objects from the `ssarp::create_SAR()` function

Usage

```
## S3 method for class 'SAR'
plot(x, ...)
```

Arguments

`x`	The SAR object that will be plotted
`...`	Parameters to pass to plot()

Value

A plot of your species-area relationship

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
occs <- find_land(occurrences = dat)
areas <- find_areas(occs = occs)
seg <- create_SAR(areas, npsi = 0)
plot(seg)
```

print.SAR *Print species-area relationship summary*

Description

Function for printing the summary of species-area relationship objects from the `ssarp::create_SAR()` function

Usage

```
## S3 method for class 'SAR'
print(x, printlen = NULL, ...)
```

Arguments

x	The SAR object of interest
printlen	Should always be NULL
...	Parameters to pass to print()

Value

The summary of your species-area relationship

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
occs <- find_land(occurrences = dat)
areas <- find_areas(occs = occs)
seg <- create_SAR(areas, npsi = 0)
print(seg)
```

---

remove_continents *Remove continents from area dataframe.*

Description

Reference a list of continental areas to remove them from the dataframe output by `ssarp::find_areas()`.

Usage

```
remove_continents(occs)
```

Arguments

occs	The dataframe that is returned by `ssarp::find_areas()`. I do not recommend using a custom dataframe for this function because it references areas given by the area database used in `ssarp::find_areas()`. If you must use a custom dataframe, please ensure that the landmass areas are in a column called "areas"

Value

A dataframe of the species name, island name, and island area (without continents)

Examples

```
# The GBIF key for the Anolis genus is 8782549
# Read in example dataset filtered from:
#  dat <- rgbif::occ_search(taxonKey = 8782549,
#                           hasCoordinate = TRUE,
#                           limit = 10000)
dat <- read.csv(system.file("extdata",
                           "ssarp_Example_Dat.csv",
                           package = "ssarp"))
occs <- find_land(occurrences = dat)
areas <- find_areas(occs = occs)
new_areas <- remove_continents(areas)
```

Index