Package: sit (via r-universe)

September 29, 2024

Type Package

Title Analyse Mark-Release-Recapture Data from Sterile Insect Technique (SIT) Field Experiments

Version 1.1.2.9015

Description Import field data about the study design such as the location and types of adult and egg traps, the release events, the trap surveys and the egg hatches. Provide functions to compute and plot various parameters of interest concerning competitiveness of sterile males, dispersal behaviour, survival rates and density of the wild male population.

License GPL-3

Encoding UTF-8

Depends R (>= 4.0)

Imports cli, dm, glue, rlang, sf, units, tidyr, knitr

Suggests spelling, tmap, covr, DiagrammeR, DiagrammeRsvg, dplyr, rmarkdown, testthat (>= 3.0.0), tibble, kableExtra, ggplot2

LazyData true

URL https://umr-astre.pages.mia.inra.fr/sit, https://forgemia.inra.fr/umr-astre/sit

BugReports https://forgemia.inra.fr/umr-astre/sit/issues

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.0

Config/testthat/edition 3

Language en-GB

Repository https://cirad-astre.r-universe.dev

RemoteUrl https://forgemia.inra.fr/umr-astre/sit

RemoteRef HEAD

RemoteSha 5565f8063e4f5a93d20279822f9538b4cd371e70

Contents

`dispersal` *Dispersal data*

Description

Compute a `data.frame` with dispersal information. I.e. age and distance from release point for each capture. Possibly adjusting for inhomogeneous trap arrangement.

Usage

```
dispersal(
  x,
  spatial_adjustment = TRUE,
  following_releases,
  following_days,
  species = NULL
)
```

Arguments

x	A sit object.
spatial_adjustment	Logical. Whether to adjust observations to account for a irregular spatial arrangement of traps.
following_releases	A sit_release object with a subset of release events or missing (default) for all point release events. Use in combination with following_days to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of NA in pop_col.
following_days	Integer or missing (default). Number of days after releases to return, if following_releases is not missing.
species	a character vector of species to be returned. Defaults to NULL, which ignores the species variable.

Details

By default, the function uses all adult surveys in the sit area of captured individuals released from **point** releases. You can control this behaviour by passing a custom sit_revents object in following_releases.

Note that the spatial adjustment is performed with the provided traps. Thus, beware of using a subset of traps only.

Value

A data.frame with dispersal data. Specifically, for each survey of sterile males, their population (colour), the trap, the age, the capture size n, the spatial weights w and the distance dist_m from the release point, in metres.

Examples

```
dispersal(sit_prototype)

yellow_release <- sit_revents(sit_prototype)[1,]
dispersal(
  sit_prototype,
  following_releases = yellow_release,
  following_days = 7L
)
```

fake_sit *Fake SIT data for testing and examples*

Description

A set of minimal data.frames with fake egg and adult surveys, point and areal release events, traps and trap-types.

Usage

```
fake_adults

fake_eggs

fake_rpoints

fake_rareal

fake_traps

fake_trap_types
```

Format

A set of 6 data.frames, some of which also sf POINT objects.

fake_adults A data.frame with 5 records from 4 days, 3 traps and 3 sterile populations + 1 wild population. The trap codes are consistent with adult traps in fake_traps.

fake_eggs A data.frame with 4 records from 3 days and 2 traps, one from the sit and one from the control areas. The trap codes are consistent with egg traps in fake_traps.

fake_rpoints A sf table with 3 point releases from 2 sites.

fake_rareal A data.frame with 2 areal releases.

fake_traps A sf table of points with 5 traps of different types. Four in the sit area and one ovitrap in the control area.

fake_trap_types A data.frame with a minimal trap type specification.

An object of class data.frame with 4 rows and 5 columns.

An object of class sf (inherits from data.frame) with 3 rows and 4 columns.

An object of class data.frame with 2 rows and 3 columns.

An object of class sf (inherits from data.frame) with 5 rows and 4 columns.

An object of class data.frame with 1 rows and 3 columns.

Source

Manually generated for testing purposes.

fertility_rate *Fertility rate*

Description

Observed fertility rate from egg surveys.

Usage

```
fertility_rate(x, species = NULL, pool = FALSE)
```

Arguments

x	Either a sit or a sit_egg_surveys object.
species	Character. Filter results for a given species or ignore the species if missing.
pool	Logical. If TRUE pools fertile and sterile counts from all traps and survey dates. Otherwise (default) yields results by trap and survey date.

Details

The sit method uses all egg surveys in the study. For a sit object x, fertility_rate(x) is thus equivalent to fertility_rate(sit_egg_surveys(x)).

Value

If pool = TRUE, a number. Otherwise, a data.frame with counts of fertile and sterile eggs by trap and survey period, with the corresponding proportion of fertile eggs.

Examples

```
fertility_rate(sit_prototype, pool = TRUE)
```

fried_index *Fried index*

Description

Index of the mating competitiveness of sterile males relative to wild males.

Usage

```
fried_index(
  sterile_wild_mr,
  natural_fertility,
  sit_fertility,
  residual_fertility
)
```

Arguments

`sterile_wild_mr`
: Non-negative number. Ratio of sterile to wild males in the target population.

`natural_fertility`
: Number between 0 and 1. Proportion of fertile eggs in a natural (wild) population.

`sit_fertility`
: Number between 0 and 1. Proportion of fertile eggs in the target population.

`residual_fertility`
: Number between 0 and 1. Proportion of fertile eggs in a completely sterile population. Also called *residual* fertility.

Details

The competitiveness γ of the sterile male individuals is defined as their relative capacity to mate with a wild female, compared to a wild male.

Thus, in a homogeneously mixed population with M_s sterile males and M_w wild males, the probability that a mating occurs with a sterile individual is

$$p_s = \frac{\gamma M_s}{M_w + \gamma M_s} = \frac{\gamma R_{sw}}{1 + \gamma R_{sw}},$$

where $R_{sw} = M_s/M_w$ is the sterile-wild male ratio. At a given sterile-wild male ratio $R_{sw} = M_s/M_w$ we observe a fertility rate H_s in the field.

Assuming a residual fertility rate H_{rs} for sterile males and a natural fertility rate H_w for wild males, the observed fertility rate H_s in the field is:

$$H_s = p_s H_{rs} + (1 - p_s) H_w = \frac{\gamma R_{sw}}{1 + \gamma R_{sw}} H_{rs} + \frac{1}{1 + \gamma R_{sw}} H_w.$$

Value

Non-negative number.

Examples

```
fried_index(
  sterile_wild_mr = 0.3,
  natural_fertility = .9,
  sit_fertility = .71,
  residual_fertility = .05
)
```

`inverse_radial_density` *Inverse radial density weights*

Description

Compute a vector of weights for a set of elements (traps), to account for inhomogeneous spatial distribution relative to a reference (release) point.

Usage

```
inverse_radial_density(x)
```

Arguments

`x`	Numeric vector of non-negative distances.

Details

The *radial density* of traps (i.e. number of traps in a ring (r, $r+dr$), divided by the surface area of the ring) is proportional to the radius under a spatially homogeneous distribution of traps, with a proportionality constant of $2/R^2$, where R is the maximum radius.

This function performs a kernel-density estimation of the radial density of the given set of traps, using a boundary correction to avoid edge effects, and returns weights calculated as the inverse of the relative density, with respect to the expected density under homogeneity.

These weights can be used to adjust trap counts by spatial arrangement, in situations where the relationship between counts and the distance from the release point is relevant.

Value

A `inverse_radial_density` object, which is a numeric vector of weights (one for each trap in `x`, in the same order), with a set of attributes used in the `plot` method.

Examples

```
## Homogeneous points
set.seed(20211129)
radial_dists_homog <- sqrt(runif(50)^2 + runif(50)^2)
ird <- inverse_radial_density(radial_dists_homog)
plot(ird)
```

`lincoln_index` *Lincoln index*

Description

Estimate the size of the wild-male population.

Usage

```
lincoln_index(released, survival_rate, age, marked_recaptured, wild_catch)
```

Arguments

`released`	Numeric. Number of sterile males released.
`survival_rate`	Numeric. Estimated survival rate.
`age`	Numeric. Number of days since release.
`marked_recaptured`	Numeric. Number of re-captured sterile males.
`wild_catch`	Numeric. Number of wild males captured.

Details

A simple estimate is obtained as follows (Thompson 2012, Ch. 18). Let the total captures at a day t be the sum of m_t marked and n_t wild mosquitoes. Assuming that the proportion of marked individuals in the sample equals that in the population of size P:

$$\frac{m_t}{m_t + n_t} = \frac{M_t}{M_t + P},$$

where $M_t = R\,S^a_t$ is the number of marked individuals captured at time t, with R the number of released adults, S the daily survival rate and a_t the number of days since release (*age*). I.e., the number of marked individuals at time t is the remaining number from those released that survived for a_t days.

The **Lincoln Index** (*a.k.a.* the **Petersen estimator**) has been used as a simple estimate of the wild **male** population size, assuming that the survival rate of an individual remains constant. Here we use a *modified* version that corrects for small samples and compensates for daily survival.

$$P_t = R\,S^{a_t}\,(n_t + 1)/(m_t + 1).$$

The values of R, n_t, m_t and t can be gathered from the adult surveys data. The calculation required the estimation of the survival rate S.

References

Thompson, Steven K. 2012. Sampling. 3rd ed. Wiley Series in Probability and Statistics. Hoboken, N.J: Wiley.

Examples

```
lincoln_index(
  released         = 1e4,
  survival_rate    = .80,
  age              = 1,
  marked_recapture = 15,
  wild_catch       = 5
)
```

`sit` *Build a* `sit` *object*

Description

Gathers data about traps, release events and field survey data into a relational data model.

Usage

```
sit(traps, release_events, adult_surveys = NULL, egg_surveys = NULL)
```

Arguments

`traps`	A `sit_traps` object built with `sit_traps()`.
`release_events`	A `sit_revents` object built with `sit_revents()`.
`adult_surveys`	A `sit_adult_surveys` object built with `sit_adult_surveys()`.
`egg_surveys`	A `sit_egg_surveys` object built with `sit_egg_surveys()`.

Value

A `sit` object that can be queried with functions from the package.

Examples

```
sit(
  sit_traps(fake_traps),
  c(sit_revents(fake_rpoints), sit_revents(fake_rareal)),
  sit_adult_surveys(fake_adults),
  sit_egg_surveys(fake_eggs)
)
```

`sit_adult_surveys` *Import or retrieve adult surveys*

Description

Import survey field data from adult traps, or retrieve the data from a `sit` object.

Usage

```
sit_adult_surveys(x, ...)

## S3 method for class 'data.frame'
sit_adult_surveys(
  x,
  trap = "trap",
  type = "type",
  survey = "survey",
  activation = "activation",
  duration = "duration",
  population = "population",
  species = "species",
  sex = "sex",
  n = "n",
  ...
)

## S3 method for class 'sit'
sit_adult_surveys(
  x,
  area = c("control", "sit"),
  trap_type = sit_trap_types(x)$label,
  following_releases,
  following_days,
  species = NULL,
  ...
)
```

Arguments

`x`	A `data.frame` with field survey data in tidy format.
`...`	Used to pass arguments to specific method.
`trap`	Identification code of the trap. This is the user's code corresponding to the `id` argument in `sit_traps()`.
`type`	Character string. Trap type.
`survey`	Character. Date and optionally the time of the survey. See details on the date format.

activation	Character. Date and optionally the time of the survey activation. See details on the date format. Either activation or duration are required.
duration	Numeric. Number of days (integer or fractional) that the trap has been functioning.
population	Character. Either wild or any of the colour codes in the release events imported with sit_revents().
species	Character. Only required if the counts of wild individuals are further categorised by species. Make sure to use the same nomenclature as in sit_revents(). In the extraction method, a character vector of species to be returned. Defaults to NULL, which ignores the species variable.
sex	Character. Either male or female.
n	Mandatory. An integer number of the surveyed individuals in the corresponding group.
area	Character vector. Either control, sit or both (default).
trap_type	Character vector. Any subset of unique(sit_trap_types(x)$label).
following_releases	A sit_release object with a subset of release events or missing (default) for all release events. Use in combination with following_days to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of NA in pop_col.
following_days	Integer or missing (default). Number of days after releases to return, if following_releases is not missing.

Details

The argument type is mandatory only if needed to uniquely identify the trap (i.e., more than one adult trap type is declared with sit_trap_types() and the same code has been used for traps of different types). If the user used unique codes for each trap, which will be checked internally, this is redundant and will be ignored in any case.

Dates must be provided in the RFC 3339 format (a variation of the ISO 8601 format), i.e., 2021-12-31 or 2019-11-23 15:00.

Value

A object of class sit_adult_surveys which can be used in sit().

Methods (by class)

- data.frame: Imports field survey data from adult traps.
- sit: Retrieve adult survey data, possibly filtering results.

See Also

sit()

Other importing: sit_egg_surveys(), sit_revents(), sit_traps()

Examples

```
ad_surv <- data.frame(
  trap = 1:3,
  survey = c("2021-04-01", "2021-08-21", "2021-08-21"),
  duration = rep(7L, 3),
  population = c("blue", "yellow", "yellow"),
  species = "aeg",
  sex = c("male", "male", "female"),
  n = 1:3
)
sit_adult_surveys(ad_surv)
```

---

sit_competitiveness *Estimate competitiveness of sterile males*

Description

Estimate competitiveness of sterile males

Usage

```
sit_competitiveness(
  x,
  following_releases = sit_revents(x, type = "areal"),
  following_days = 7,
  residual_fertility = 0.01,
  species = NULL
)
```

Arguments

x	A sit object.
following_releases	A sit_release object with a subset of release events or missing (default) for all release events. Use in combination with following_days to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of NA in pop_col.
following_days	Integer or missing (default). Number of days after releases to return, if following_releases is not missing.
residual_fertility	Numeric. Default: 0.05.
species	Character. Filter results for a given species or ignore the species if missing.

Value

A numeric value.

Examples

```
sit_competitiveness(sit_prototype)
```

`sit_diffusion` *Diffusion*

Description

Diffusion

Usage

```
sit_diffusion(
  x,
  pool = FALSE,
  spatial_adjustment = TRUE,
  following_releases,
  following_days,
  species = NULL
)
```

Arguments

`x`	A `sit` object.
`pool`	Logical. Whether to aggregate data from released populations.
`spatial_adjustment`	Logical. Whether to adjust observations to account for a irregular spatial arrangement of traps.
`following_releases`	A `sit_release` object with a subset of release events or missing (default) for all point release events. Use in combination with `following_days` to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of `NA` in `pop_col`.
`following_days`	Integer or missing (default). Number of days after releases to return, if `following_releases` is not missing.
`species`	a character vector of species to be returned. Defaults to `NULL`, which ignores the species variable.

Value

A named numeric vector with one value for each requested `level`.

Examples

```
(diff_prototype <- sit_diffusion(sit_prototype))
plot(diff_prototype)
```

`sit_egg_surveys` *Import or retrieve egg surveys*

Description

Import survey field data from egg traps, or retrieve the data from a `sit` object.

Usage

```
sit_egg_surveys(x, ...)

## S3 method for class 'data.frame'
sit_egg_surveys(
  x,
  trap = "trap",
  type = "type",
  survey = "survey",
  activation = "activation",
  duration = "duration",
  species = "species",
  fertile = "fertile",
  n = "n",
  ...
)

## S3 method for class 'sit'
sit_egg_surveys(
  x,
  area = c("control", "sit"),
  trap_type = sit_trap_types(x)$label,
  following_releases,
  following_days,
  species = NULL,
  ...
)
```

Arguments

`x`	A `data.frame` with field survey data in tidy format.
`...`	Used to pass arguments to specific method.
`trap`	Identification code of the trap. This is the user's code corresponding to the `id` argument in `sit_traps()`.
`type`	Character string. Trap type.
`survey`	Character. Date and optionally the time of the survey. See details on the date format.

`activation`	Character. Date and optionally the time of the survey activation. See details on the date format. Either `activation` or `duration` are required.
`duration`	Numeric. Number of days (integer or fractional) that the trap has been functioning.
`species`	Character.
`fertile`	Logical. Whether `n` represents counts of **fertile** eggs (`TRUE`) of **sterile** eggs (`FALSE`).
`n`	Mandatory. An integer number of the surveyed individuals in the corresponding group.
`area`	Character vector. Either `control`, `sit` or both (default).
`trap_type`	Character vector. Any subset of `unique(sit_trap_types(x)$label)`.
`following_releases`	A `sit_release` object with a subset of release events or missing (default) for all release events. Use in combination with `following_days` to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of `NA` in `pop_col`.
`following_days`	Integer or missing (default). Number of days after releases to return, if `following_releases` is not missing.

Details

The argument `type` is mandatory only if needed to uniquely identify the trap (i.e., more than one egg trap type is declared with `sit_trap_types()` and the same code has been used for traps of different types). If the user used unique codes for each trap, which will be checked internally, this is redundant and will be ignored in any case.

Dates must be provided in the RFC 3339 format (a variation of the ISO 8601 format), i.e., `2021-12-31` or `2019-11-23 15:00`.

Value

A object of class `sit_egg_surveys` which can be used in `sit()`.

Methods (by class)

- `data.frame`: Imports field survey data from egg traps.
- `sit`: Retrieve egg survey data, possibly filtering results.

See Also

`sit()`

Other importing: `sit_adult_surveys()`, `sit_revents()`, `sit_traps()`

Examples

```
egg_surv <- data.frame(
  trap = 1:3,
  survey = c("2021-04-01", "2021-08-21", "2021-08-21"),
  duration = rep(7L, 3),
  fertile = c(TRUE, FALSE, TRUE),
  n = 1:3
)
sit_egg_surveys(egg_surv)
```

`sit_flight_range` *Flight Range*

Description

Flight Range

Usage

```
sit_flight_range(
  x,
  pool = FALSE,
  levels = c(50, 90),
  spatial_adjustment = TRUE,
  following_releases,
  following_days,
  species = NULL
)
```

Arguments

`x`	A `sit` object.
`pool`	Logical. Whether to aggregate data from released populations.
`levels`	Numeric vector with values between 0 and 100. Requested levels of flight range, in percentage.
`spatial_adjustment`	Logical. Whether to adjust observations to account for a irregular spatial arrangement of traps.
`following_releases`	A `sit_release` object with a subset of release events or missing (default) for all point release events. Use in combination with `following_days` to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of `NA` in `pop_col`.
`following_days`	Integer or missing (default). Number of days after releases to return, if `following_releases` is not missing.
`species`	a character vector of species to be returned. Defaults to `NULL`, which ignores the species variable.

Value

A named numeric vector with one value for each requested `level`.

Examples

```
sit_flight_range(sit_prototype)
```

`sit_mdt` *Mean Distance Travelled and Mean Dispersal Distance*

Description

Mean Distance Travelled and Mean Dispersal Distance

Usage

```
sit_mdt(
  x,
  by = c("population", "age"),
  spatial_adjustment = TRUE,
  following_releases,
  following_days,
  species = NULL
)
```

Arguments

`x`	A `sit` object.
`by`	Character vector. Either 'population', or 'age', both (default), or NULL.
`spatial_adjustment`	Logical. Whether to adjust observations to account for a irregular spatial arrangement of traps.
`following_releases`	A `sit_release` object with a subset of release events or missing (default) for all point release events. Use in combination with `following_days` to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of `NA` in `pop_col`.
`following_days`	Integer or missing (default). Number of days after releases to return, if `following_releases` is not missing.
`species`	a character vector of species to be returned. Defaults to `NULL`, which ignores the species variable.

Value

By default (i.e. `by = c('population', 'age')`), a table with MDT computed for each released population at each *age* (i.e., number of days since release). If `by` is either one of the grouping variables, the results will be presented for each value of the corresponding variable. Finally, if `by = NULL`, all populations and ages are pooled together and the function returns a single number.

Examples

```
sit_mdt(sit_prototype)
sit_mdt(sit_prototype, by = 'population')
sit_mdt(sit_prototype, by = 'age')
sit_mdt(sit_prototype, by = NULL)
```

---

sit_prototype *A prototypical* `sit` *object with fake data*

Description

A `sit` object with realistic (but fake) data for demonstration purposes.

Usage

```
sit_prototype
```

Format

A `sit` object with 3 point releases, all from the same location, and 1 areal release; 21 adult and 14 traps in the study area, 7 ovitraps in a control area. All ovitraps lacking coordinates. Adult traps surveyed daily for 36 consecutive days starting the day after the first release. Ovitraps from both areas surveyed weekly.

Source

Manually generated for demonstration purposes.

---

sit_revents *Import or extract release events*

Description

Provide a `sf` object with a POINT at each record corresponding to a **release point** or a `data.frame` for a **areal release**, specifying which variables contain the required information. Or, extract the release events used in a `sit` object.

Usage

```
sit_revents(x, ...)

## S3 method for class 'sf'
sit_revents(x, date = "date", colour = "colour", n = "n", species = NULL, ...)

## S3 method for class 'data.frame'
sit_revents(x, date = "date", colour = "colour", n = "n", species = NULL, ...)

## S3 method for class 'sit_revents'
c(...)

## S3 method for class 'sit'
sit_revents(x, type = c("point", "areal"), ...)
```

Arguments

`x`	Object of class `sf` with POINT elements to be *imported*, or a `data.frame` with information about areal releases, or object of class `sit` to *extract* release events from.
`...`	Used to pass arguments to specific method.
`date`	Character string representing a date and optionally the time, in the RFC 3339 format (a variation of the ISO 8601 format), i.e., `2021-12-31` or `2019-11-23 15:00`.
`colour`	Character string. Colour of the release.
`n`	Numeric. Natural number. Number of individuals released.
`species`	Optional character. Released species.
`type`	Character vector. Which *types* of release events to retrieve. Either `point` or `areal` or both (default).

Value

A object of class `sit_revents` which can be used in `sit()`. `sit_revents` objects can be concatenated together with `c()`, in order to import a mixture of point and areal releases into a `sit` object. See examples.

Methods (by class)

- `sf`: Imports point release data.
- `data.frame`: Imports areal release data.
- `sit_revents`: Concatenate release events.
- `sit`: Extracts release events.

See Also

`sit()`

Other importing: `sit_adult_surveys()`, `sit_egg_surveys()`, `sit_traps()`

Examples

```
point_releases <- sit_revents(
  sf::st_as_sf(
    data.frame(
      x = 1:3,
      y = 3:1,
      date = c("2019-11-25", "2019-12-01", "2019-12-13"),
      colour = c("yellow", "red", "blue"),
      n = 1e4
    ),
    coords = c("x", "y")
  )
)

areal_release <- sit_revents(
  data.frame(
    x = 1, y = 1,
    date = "2019-12-21",
    colour = "pink",
    n = 1e4
  )
)

(c(point_releases, areal_release)) # also a `sit_revent` object
```

---

sit_survival *Survival statistics*

Description

Compute the *Probability of Daily Survival* (PDS), the *Average Life Expectancy* ALE, *Recapture Rate* RR, *Survival Rate* SR), by population (by default, unless pool = TRUE)

Usage

```
sit_survival(
  x,
  pool = FALSE,
  spatial_adjustment = TRUE,
  following_releases,
  following_days,
  species = NULL
)
```

Arguments

x	A sit object.

pool Logical. Whether to aggregate data from released populations.

spatial_adjustment
Logical. Whether to adjust observations to account for a irregular spatial arrangement of traps.

following_releases
A sit_release object with a subset of release events or missing (default) for all point release events. Use in combination with following_days to filter surveys of the target populations within a given number of days after release. Note that counts of wild populations will always be included in the results with a value of NA in pop_col.

following_days Integer or missing (default). Number of days after releases to return, if following_releases is not missing.

species a character vector of species to be returned. Defaults to NULL, which ignores the species variable.

Value

A sit_survival object, which is a data.frame with additional attributes which can be plotted.

Examples

```
(surv_prototype <- sit_survival(sit_prototype))
plot(surv_prototype)
```

---

sit_traps *Import or extract trap information*

Description

Provide a sf object with a POINT at each record corresponding to a trap location, specifying which variables contain the required information. Or define a set of traps without spatial coordinates. Or, extract the trap data used in a sit object.

Usage

```
sit_traps(x, ...)

## S3 method for class 'sf'
sit_traps(
  x,
  id = "id",
  type = "type",
  area = "area",
  label,
  trap_types = sit_trap_types(),
  ...
```

```
)

## S3 method for class 'character'
sit_traps(x, type, area, trap_types = sit_trap_types(), ...)

## S3 method for class 'numeric'
sit_traps(x, ...)

## S3 method for class 'sit_traps'
c(...)

## S3 method for class 'sit'
sit_traps(
  x,
  type = sit_trap_types(x)$label,
  stage = unique(sit_trap_types(x)$stage),
  area = unique(x$traps$area),
  ...
)
```

Arguments

x	Object of class `sf` with POINT elements to be *imported*, or a character-coerced vector *trap codes* to be defined, or an object of class `sit` to *extract* trap information from.
...	Used to pass arguments to specific method.
id	Character. Variable name with a custom code for the trap. Jointly with `type`, identifies the trap uniquely.
type	Character. Variable name with **valid** values of trap types (i.e., any of `trap_types$label`). For the extraction method, a character vector of trap types to filter upon.
area	Character. Variable name with values either `control` or `sit`. For the extraction method, a character vector of areas to filter upon.
label	Optional character. Variable name with custom text associated to the traps.
trap_types	Table of trap types, created with `sit_trap_types()`.
stage	Character vector of stages (i.e., `adult` or `egg`) to filter upon extraction.

Value

A object of class `sit_traps` which can be used in `sit()`.

Methods (by class)

- `sf`: Imports spatialised trap data
- `character`: Imports non-spatialised trap data
- `numeric`: Imports non-spatialised trap data
- `sit_traps`: Combine traps.
- `sit`: Extracts trap data

See Also

sit()

Other importing: sit_adult_surveys(), sit_egg_surveys(), sit_revents()

Examples

```
## Build spatialised traps from a `sf` object
traps_sf <- sf::st_as_sf(
  data.frame(
    x = 1:3,
    y = 3:1,
    "Trap.Id" = letters[1:3],
    Type = "BGS",
    area = "sit"
  ),
  coords = c("x", "y")
)
sit_traps(traps_sf, id = "Trap.Id", type = "Type", area = "area")

## Build non-spatialised traps from a vector of codes
sit_traps(paste0("C", 1:5), area = 'control', type = 'OVT')

## Retrieve traps in a `sit` object
## sit_traps(sit_prototype)
```

---

sit_trap_types *Import or extract trap types*

Description

Extracts the table of trap types used in a sit experiment, or defines a table of trap types.

Usage

```
sit_trap_types(x)

## S3 method for class 'sit'
sit_trap_types(x)

## S3 method for class 'data.frame'
sit_trap_types(x)
```

Arguments

x	A sit object, a data.frame-like object or empty.

Details

Returns the table of trap types used in the `sit` object `x`.

Defines a table of trap types by passing a `data.frame` with variables `name`, `label`, `stage` (either `egg` or `adult`) and optionally `description`. Any other variable is ignored with a warning.

Returns the default table of trap types by calling `sit_trap_types()` without any argument.

Check the default trap types with `sit_trap_types()`. You can store the table into an object and make edits as for a regular `data.frame`. E.g. add another trap type, or edit labels or descriptions. You can use the edited object as an input to the `trap_types` arguments in `sit_traps()`.

Value

Table of trap types used in the `sit` experiment `x`. If `x` is a `data.frame`, returns it as a table of trap types after some verifications. If `x` is missing, returns the current table of trap types.

Methods (by class)

- `sit`: Extract trap data
- `data.frame`: Import trap data

Examples

```
sit_trap_types()  # default trap types.

# Define new trap types
my_traps <- data.frame(name = "My New Trap", label = "MNT", stage = "adult")
sit_trap_types(my_traps)
```

`sit_wild_size` *Size of the Wild Population*

Description

Estimate the size of the wild population, via the Lincoln Index.

Usage

```
sit_wild_size(x, pool = FALSE, species = NULL)
```

Arguments

`x`	A `sit` object.
`pool`	Logical. Whether to aggregate data from released populations.
`species`	a character vector of species to be returned. Defaults to `NULL`, which ignores the species variable.

Details

This provides multiple estimates of the population size, one for each age t and for each released population, which can be averaged to compute a final estimate based on all the available data.

The calculation needs to be split by released population. First, because survival rates could be different among populations, but also because the number n_t of captured wild males at a given date can be compared against the number of captured sterile males m_t at different ages.

Value

A `sit_wild_size` object.

Examples

```
sit_wild_size(sit_prototype)
```

`sterile_wild_male_ratio`
Sterile-Wild male ratio

Description

Observed ratio between sterile and wild male individuals from adult surveys.

Usage

```
sterile_wild_male_ratio(x, pool)
```

Arguments

`x`	Either a `sit` or a `sit_adult_surveys` object.
`pool`	Logical. If `TRUE` pools sterile and wild counts from all traps and survey dates. Otherwise (default) yields results by trap and survey date.

Details

The `sit` method uses all adult surveys from the study (`sit`) area. For a `sit` object `x`, `sterile_wild_male_ratio(x)` is thus equivalent to `sterile_wild_male_ratio(sit_adult_surveys(x, area = 'sit'))`.

Value

If `pool = TRUE`, a number. Otherwise, a data.frame with counts of sterile and wild males by trap and survey period, with the corresponding ratio.

Examples

```
sterile_wild_male_ratio(sit_prototype, pool = TRUE)
```

`survey_ages` *Age of captured sterile males*

Description

Compute the *age* (i.e. number of days since release to capture) of released sterile males.

Usage

```
survey_ages(surveys, releases)
```

Arguments

`surveys`	A `sit_adult_surveys` object.
`releases`	A `sit_revents` object with target releases.

Value

A data.frame, with a similar structure as `surveys` but with captures of the sterile males from the target releases only, and with additional variables for the release date and the age of the individual at the moment of the survey in number of days.

Examples

```
head(
  survey_ages(sit_adult_surveys(sit_prototype), sit_revents(sit_prototype))
)
```

Index