Package 'epiphy'

May 8, 2026

Type Package

Title Analysis of Plant Disease Epidemics

Version 0.5.0

Description A toolbox to make it easy to analyze plant disease epidemics. It provides a common framework for plant disease intensity data recorded over time and/or space. Implemented statistical methods are currently mainly focused on spatial pattern analysis (e.g., aggregation indices, Taylor and binary power laws, distribution fitting, SADIE and 'mapcomp' methods). See Laurence V. Madden, Gareth Hughes, Franck van den Bosch (2007) <doi:10.1094/9780890545058> for further information on these methods. Several data sets that were mainly published in plant disease epidemiology literature are also included in this package.

License MIT + file LICENSE

URL https://github.com/chgigot/epiphy, https://chgigot.github.io/epiphy/

BugReports https://github.com/chgigot/epiphy/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

LinkingTo Rcpp

Depends R (>= 4.0)

Imports stats, methods, utils, grDevices, ggplot2, transport, msm, pbapply, Rcpp

Suggests magrittr, dplyr, tidyr, spdep, emdist, vegan, MASS, emdbook, knitr, rmarkdown

Collate 'RcppExports.R' 'betabinom.R' 'data.R' 'utils.R' 'mle-factory.R' 'epiphy.R' 'intensity-classes.R' 'distr-fitting.R' 'indices.R' 'mapcomp.R' 'power-law.R' 'sadie.R' 'spatial-hier.R'

VignetteBuilder knitr

NeedsCompilation yes

Author Christophe Gigot [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0002-1748-2902>),
Adam H. Sparks [ctb],
Katrin Leinweber [ctb]

Maintainer Christophe Gigot <ch.gigot@gmail.com>

Repository CRAN

Date/Publication 2023-11-16 11:20:10 UTC

Contents

a2a *Easily switch between different power law formulations.*

Description

a2a was designed to avoid headaches that are likely to occur when working with different formulations of the binomial power law analysis.

Usage

```
a2a(x, ...)

## S3 method for class 'numeric'
a2a(
  x,
  slope,
  n,
  from = c("Ai", "ai", "AI", "aI"),
  to = c("Ai", "ai", "AI", "aI"),
  ...
)

## S3 method for class 'list'
a2a(x, to = c("Ai", "ai", "AI", "aI"), ...)
```

Arguments

x	Intercept parameter to be converted or a named list with the parameter to be converted ("Ai", "ai", "AI" or "aI"), the slope ("slope"), and the number of individual per sampling unit ("n").
...	Additional arguments to be passed to other methods.
slope	Slope parameter.
n	Number of individuals per sampling unit.
from	Kind of the input intercept parameter ("Ai", "ai", "AI" or "aI").
to	Desired kind for the ouput intercept parameter ("Ai", "ai", "AI" or "aI").

Details

The binomial power law can be expressed as: $s_y^2 = (intercept)(s_{bin}^2)^b$. But different forms of (intercept) are possible depending on the formulation of the binomial power law.

	Ai	ai	AI	aI
Ai	1	n^b	n^(2(b-1))	n^(b-2)
ai	n^(-b)	1	n^(b-2)	n^(-2)
AI	n^(2(1-b))	n^(2-b)	1	n^(-b)

aI n^(2-b) n^2 n^b 1

Value

A numeric vector.

Examples

```
# Values from the power_law() example:
Ai    <- 38.6245
slope <- 1.9356
n     <- 9

# Usual function call syntax:
a2a(Ai, slope, n, from = "Ai", to = "ai")

# Other syntaxes:
inputs <- list(Ai = Ai, slope = slope, n = n)
a2a(inputs, "ai")
require(magrittr)
inputs %>% a2a("ai")
```

---

agg_index *Several aggregation indices.*

Description

This function can compute different aggregation indices. See "Details" section for more information about the available indices.

Usage

```
agg_index(
  x,
  method = c("fisher", "lloyd", "morisita"),
  flavor = c("count", "incidence"),
  n = NULL,
  ...
)
```

Arguments

x	A numeric vector or a count/incidence object.
method	The name of the method to be used. "fisher" method is used by default. See details below.
flavor	Which flavor of this index must be calculated ("count" or "incidence")?

n	Number of individuals per sampling unit. If n is provided, the "incidence" flavor is calculated whatever the value of flavor. Note that current implementation only deals with equal size sampling units.
...	Additional arguments to be passed to other methods.

Details

There are currently three implemented methods to compute indices of aggregation.

fisher: Fisher's index of aggregation. In case of a count, this index corresponds to the ratio of the observed variance to the observed mean, and this is why this index is also known as the variance to mean ratio. For a binary variable, a similar index can be calculated using instead the ratio of the observed variance to the theoretical variance if data follow a binomial law (i.e. a reference distribution for a random pattern of individuals within sampling units).

lloyd: Lloyd's index of patchiness. The value of this index increases with the degree of aggregation. Note that Lloyd's mean crowding can also be returned if type = "mean-crowding" is provided as parameter.

morisita: Morisita's coefficient of dispersion. This index can be computed for either count or incidence data, but its interpretation can be uncertain.

Values of Fisher's and Lloyd's indices can be interpreted as follows:

- index < 1: uniform pattern;
- index = 1: random pattern;
- index > 1: aggregated pattern.

The following table gives information about the applicability of the various methods.

	count	incidence	severity
fisher	+	+	-
lloyd	+	-	-
morisita	+	+	-

where + means implemented, and -, not implemented (or not possible). At the moment, there is no index of aggregation for severity data.

Value

An object of class agg_index, which is a list containing the following components:

index	The value of the index.
name	The name of the index.
flavor	The flavor of the calculated index ("count" or "incidence").
N	The number of sampling units.
n	The number of individuals in each sampling unit (if relevant).

References

Fisher RA. 1925. Statistical methods for research workers. Oliver and Boyd, Edinburgh.

Lloyd M. 1967. Mean crowding. The Journal of Animal Ecology 36, 1–30.

Morisita M. 1962. Iδ-Index, a measure of dispersion of individuals. Researches on Population Ecology 4, 1–7. doi:10.1007/BF02533903

Madden LV, Hughes G. 1995. Plant disease incidence: Distributions, heterogeneity, and temporal analysis. Annual Review of Phytopathology 33(1): 529–564. doi:10.1146/annurev.py.33.090195.002525

See Also

vegdist in **vegan** package.

Examples

```
# Count flavor of Fisher's index:
my_fisher_count <- agg_index(aphids$i)
my_fisher_count

# And incidence flavor of Fisher's index:
my_fisher_incidence <- agg_index(tobacco_viruses$i, n = tobacco_viruses$n)
my_fisher_incidence

# Either standard R or epiphy idioms can be used:
identical(my_fisher_count, agg_index(count(aphids)))
identical(my_fisher_incidence, agg_index(incidence(tobacco_viruses)))

# Lloyd's index (only for count data):
agg_index(aphids$i, method = "lloyd")
# Lloyd's mean crowding:
agg_index(aphids$i, method = "lloyd", type = "mean-crowding")

# Count flavor of Morisita's index:
agg_index(aphids$i,  method = "morisita")
# Incidence flavor of Morisita's index:
agg_index(tobacco_viruses$i, n = tobacco_viruses$n, method = "morisita")
```

---

aphids *Counts of aphids.*

Description

Counts of 554 aphids of the species Sitobion avenae, sampled on 28 June 1996 in a 250 x 180-m field of winter wheat near Wimborne, Dorset, UK. The 63 sampling units, made of the inspection of five tillers each, were located on a 9 x 7 rectangular grid at intervals of 30 m.

Usage

```
aphids
```

Format

A data frame with 63 rows and 3 variables:

[, 1:2]	x,y	Grid spatial coordinates.
[, 3:4]	xm,ym	Metric spatial coordonates.
[, 5]	i	Counts of aphids.

Source

Perry JN, Winder L, Holland JM, Alston RD. 1999. Red-blue plots for detecting clusters in count data. Ecology Letters 2, 106-13. doi:10.1046/j.14610248.1999.22057.x

---

`arthropods` *Counts of arthropods.*

Description

A sampling unit was made of a pitfall to collect arthropods in a field of organic winter wheat, near Wimborne, Dorset, UK in 1996. The sampling units were located on a 9 x 7 rectangular grid at intervals of 30 m. There were six sampling dates.

Usage

```
arthropods
```

Format

A data frame with 378 rows and 4 variables:

[, 1:2]	x,y	Grid spatial coordinates.
[, 3:4]	xm,ym	Metric spatial coordonates.
[, 5]	t	Sampling date. 1: 7 Jun, 2: 14 Jun, 3: 21 Jun, 4: 28 Jun, 5: 5 Jul, 6: 12 Jul 1996.
[, 6]	i	Counts of arthropods.

Source

Holland JM, Winder L, Perry JN. 1999. Arthropod prey of farmland birds: Their spatial distribution within a sprayed field with and without buffer zones. Aspects of Applied Biology 54: 53-60.

`as.data.frame.intensity` *Coerce to a data frame.*

Description

Functions to coerce an `intensity` object to a data frame.

Usage

```
## S3 method for class 'intensity'
as.data.frame(
  x,
  row.names = NULL,
  optional = FALSE,
  ...,
  stringsAsFactors = FALSE
)
```

Arguments

`x`	An `intensity` object.
`row.names`	`NULL` or a character vector giving the row names for the data frame. Missing values are not allowed.
`optional`	logical. If `TRUE`, setting row names and converting column names (to syntactic names: see `make.names`) is optional. Note that all of R's **base** package `as.data.frame()` methods use `optional` only for column names treatment, basically with the meaning of `data.frame(*, check.names = !optional)`. See also the `make.names` argument of the `matrix` method.
`...`	additional arguments to be passed to or from methods.
`stringsAsFactors`	logical: should the character vector be converted to a factor?

Value

A data frame.

Examples

```
my_data <- incidence(tomato_tswv$field_1929)
head(as.data.frame(my_data))
```

BetaBinomial *The beta-binomial distribution.*

Description

Density, distribution function, quantile function and random generation for the beta-binomial distribution with parameters `size`, `prob`, `theta`, `shape1`, `shape2`. This distribution corresponds to an overdispersed binomial distribution.

Usage

```
dbetabinom(x, size, prob, theta, shape1, shape2, log = FALSE)

pbetabinom(
  q,
  size,
  prob,
  theta,
  shape1,
  shape2,
  lower.tail = TRUE,
  log.p = FALSE
)

qbetabinom(
  p,
  size,
  prob,
  theta,
  shape1,
  shape2,
  lower.tail = TRUE,
  log.p = FALSE
)

rbetabinom(n, size, prob, theta, shape1, shape2)
```

Arguments

`x`, `q`	Vector of quantiles.
`size`	Number of trials.
`prob`	Probability of success on each trial.
`theta`	Aggregation parameter (theta = 1 / (shape1 + shape2)).
`shape1`, `shape2`	Shape parameters.
`log`, `log.p`	Logical; if TRUE, probabilities p are given as log(p).

lower.tail	Logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
p	Vector of probabilities.
n	Number of observations.

Details

Be aware that in this implementation theta = 1 / (shape1 + shape2). prob and theta, or shape1 and shape2 must be specified. if theta = 0, use *binom family instead.

Value

dbetabinom gives the density, pbetabinom gives the distribution function, qbetabinom gives the quantile function and rbetabinom generates random deviates.

See Also

dbetabinom in the package **emdbook** where the definition of theta is different.

Examples

```
# Compute P(25 < X < 50) for X following the Beta-Binomial distribution
# with parameters size = 100, prob = 0.5 and theta = 0.35:
sum(dbetabinom(25:50, size = 100, prob = 0.5, theta = 0.35))

# When theta tends to 0, dbetabinom outputs tends to dbinom outputs:
sum(dbetabinom(25:50, size = 100, prob = 0.5, theta = 1e-7))
sum(dbetabinom(25:50, size = 100, shape1 = 1e7, shape2 = 1e7))
sum(dbinom(25:50, size = 100, prob = 0.5))

# Example of binomial and beta-binomial frequency distributions:
n   <- 15
q   <- 0:n
p1  <- dbinom(q, size = n, prob = 0.33)
p2  <- dbetabinom(q, size = n, prob = 0.33, theta = 0.22)
res <- rbind(p1, p2)
dimnames(res) <- list(c("Binomial", "Beta-binomial"), q)
barplot(res, beside = TRUE, legend.text = TRUE, ylab = "Frequency")

# Effect of the aggregation parameter theta on probability density:
thetas <- seq(0.001, 2.5, by = 0.001)
density1 <- rep(sum(dbinom(25:50, size = 100, prob = 0.5)), length(thetas))
density2 <- sapply(thetas, function(theta) {
    sum(dbetabinom(25:50, size = 100, prob = 0.5, theta = theta))
})
plot(thetas, density2, type = "l",
     xlab = expression("Aggregation parameter ("*theta*")"),
     ylab = "Probability density between 25 and 50 (size = 100)")
lines(thetas, density1, lty = 2)
```

`calpha.test` *C(alpha) test.*

Description

The C(alpha) test is a test of the binomial distribution against the alternative of the beta-binomial distribution.

Usage

```
calpha.test(x, ...)

## S3 method for class 'fisher'
calpha.test(x, ...)
```

Arguments

`x`	The output of the `agg_index` function with `method = "fisher"` as parameter.
`...`	Not yet implemented.

Details

It is based on calculation of a test statistic, z, that has an asymptotic standard normal distribution under the null hypothesis. It is one-sided (in the way that the alternative is aggregation, not just "non-randomness"), thus with a confidence level of 95 1.64. When all the sampling units contain the same total number of individuals, n, the test statistic is calculated from:

z = (n(N - 1)I - Nn)/(2Nn(n - 1))^(1/2)

where N is the number of sampling units, and I, Fisher's index of aggregation for incidence data.

Value

Same kind of object as the one returns by the stats `chisq.test` function for example.

References

Neyman J. 1959. Optimal asymptotic tests of composite statistical hypotheses. In: Probability and Statistics, 213-234. Wiley, New York.

Tarone RE. 1979. Testing the goodness of fit of the binomial distribution. Biometrika, 66(3): 585-590.

See Also

`chisq.test`, `z.test`

Examples

```
# For incidence data:
my_incidence <- incidence(tobacco_viruses)
my_fisher <- agg_index(my_incidence, method = "fisher")
calpha.test(my_fisher)
```

`chisq.test` *Chi-squared test.*

Description

Performs chi-squared tests for Fisher's aggregation indices (computed with either count or incidence data). If another kind of data is provided, the R standard `chisq.test` function is called.

Usage

```
chisq.test(x, ...)

## Default S3 method:
chisq.test(x, ...)

## S3 method for class 'fisher'
chisq.test(x, ...)
```

Arguments

`x`	Either the output of the `agg_index` function with `method = "fisher"` as parameter, or another R object. In the latter case, stats::`chisq.test` is called.
`...`	Further arguments to be passed to stats::`chisq.test`.

Details

Under the null hypothesis for Fisher's aggregation index (index = 1, i.e. a random pattern is observed), (N - 1)*index follows a chi-squared distribution with N - 1 degrees of freedom. N is the number of sampling units.

Value

Same kind of object as the one returns by the stats `chisq.test` function.

References

For count and incidence data:

Madden LV, Hughes G. 1995. Plant disease incidence: Distributions, heterogeneity, and temporal analysis. Annual Review of Phytopathology 33(1): 529–564. doi:10.1146/annurev.py.33.090195.002525

Patil GP, Stiteler WM. 1973. Concepts of aggregation and their quantification: a critical review with some new results and applications. Researches on Population Ecology, 15(1): 238-254.

See Also

calpha.test, z.test

Examples

```
# For incidence data:
my_incidence <- incidence(tobacco_viruses)
my_fisher <- agg_index(my_incidence, method = "fisher")
chisq.test(my_fisher)
```

---

citrus_ctv *Incidence of citrus tristeza virus (CTV) disease in three fields.*

Description

CTV incidence data for three orchards in eastern Spain reported for consecutive years.

Usage

```
citrus_ctv
```

Format

There are three data frames:

- IVI3and4: A data frame with 864 rows and 5 variables.
- IVI6and7: A data frame with 648 rows and 5 variables.
- El_Realengo: A data frame with 2000 rows and 5 variables.

The structure is the same for all the data frames:

[, 1:2]	x,y	Grid spatial coordinates.
[, 3]	t	Year of disease assessments.
[, 4]	i	Disease incidence. 0: Healthy, 1: Diseased.
[, 5]	n	Sampling unit size. n = 1 means that the sampling unit size is the plant.

Details

Both IVI3and4 and IVI6and7 orchards consisted of 216 trees each of Washington navel orange on Troyer citrange planted in 1978 on a 2 x 6-m spacing. El_Realengo orchard consisted of 400 Marsh seedless grapefruit on Troyer citrange planted in 1973 on a 5.5 x 5.5-m spacing.

Source

Gottwald TR, Cambra M, Moreno P, Camarasa E, Piquer J. 1996. Spatial and temporal analyses of citrus tristeza virus in eastern Spain. Phytopathology 86, 45–55.

Gibson GJ. 1997. Investigating mechanisms of spatiotemporal epidemic spread using stochastic models. Phytopathology 87, 139-46. doi:10.1094/PHYTO.1997.87.2.139

clump *Regroup observational data into even clumps of individuals.*

Description

This function provides a easy way to regroup recorded data into groups of same number of individuals.

Usage

```
clump(object, ...)

## S3 method for class 'intensity'
clump(object, unit_size, fun = sum, inclusive_unspecified = FALSE, ...)
```

Arguments

object	An intensity object.
...	Additional arguments to be passed to fun.
unit_size	Size of a group unit. It must be a named vector, with names corresponding to non-observational variables (i.e. space and time variables). If the size of a variable in the data set is not a multiple of the provided value in unit_size, some sampling units (the last ones) will be dropped so that clumps of individuals remain even throughout the data set.
fun	Function used to group observational data together.
inclusive_unspecified	Not yet implemented. Do unspecified mapped variables (different from i and n) need to be included into the bigger possible sampling unit (TRUE) or splited into as many sampling units as possible (FALSE, default)?

Value

An intensity object.

Examples

```
my_incidence <- incidence(tomato_tswv$field_1929)
plot(my_incidence, type = "all")

# Different spatial size units:
my_incidence_clumped_1 <- clump(my_incidence, unit_size = c(x = 3, y = 3))
plot(my_incidence_clumped_1, type = "all")

my_incidence_clumped_2 <- clump(my_incidence, unit_size = c(x = 4, y = 5))
plot(my_incidence_clumped_2, type = "all")

# To get mean disease incidence for each plant over the 3 scoring dates:
```

```
my_incidence_clumped_3 <- clump(my_incidence, unit_size = c(t = 3), fun = mean)
plot(my_incidence_clumped_3)
```

`codling_moths` *Count of codling moth larvae.*

Description

Codling moth diapausing larvae were collected in an apple orchard in south-eastern France. Larvae were caught on strip traps wrapped around tree trunks in July 2008 and collected the following October. 30 traps were used.

Usage

```
codling_moths
```

Format

A data frame with 30 rows and 3 variables:

[, 1:2]	x,y	Metric spatial coordonates.
[, 3]	i	Counts of codling moth larvae.

Source

Lavigne C, Ricci B, Franck P, Senoussi R. 2010. Spatial analyses of ecological count data: A density map comparison approach. Basic and Applied Ecology 11: 734-42. doi:10.1016/j.baae.2010.08.011

`dogwood_anthracnose` *Incidence of dogwood anthracnose.*

Description

Incidence data from the Dogwood Anthracnose Impact Assessment Program for 1990 and 1991, in the Southeast of the USA, reported by Zarnoch et al (1995). Only plots with exactly n = 10 dogwood trees are present in the data set (168 and 161 plots in 1990 and 1991, respectively).

Usage

```
dogwood_anthracnose
```

Format

A data frame with 329 rows and 3 variables:

[, 1]	t	Year of disease assessments (1990 or 1991)..
[, 2]	i	Number of diseased plants (from 0 to 10).
[, 3]	n	Sampling unit size. Here, n = 10 plants per sampling unit (or plot).

Source

Zarnoch SJ, Anderson RL, Sheffield RM. 1995. Using the β-binomial distribution to characterize forest health. Canadian journal of forest research 25, 462–469.

`fit_two_distr` *Maximum likelihood fitting of two distributions and goodness-of-fit comparison.*

Description

Different distributions may be used depending on the kind of provided data. By default, the Poisson and negative binomial distributions are fitted to count data, whereas the binomial and beta-binomial distributions are used with incidence data. Either Randomness assumption (Poisson or binomial distributions) or aggregation assumption (negative binomial or beta-binomial) are made, and then, a goodness-of-fit comparison of both distributions is made using a log-likelihood ratio test.

Usage

```
fit_two_distr(data, ...)

## Default S3 method:
fit_two_distr(data, random, aggregated, ...)

## S3 method for class 'count'
fit_two_distr(
  data,
  random = smle_pois,
  aggregated = smle_nbinom,
  n_est = c(random = 1, aggregated = 2),
  ...
)

## S3 method for class 'incidence'
fit_two_distr(
  data,
  random = smle_binom,
  aggregated = smle_betabinom,
  n_est = c(random = 1, aggregated = 2),
```

```
  ...
)
```

Arguments

data	An intensity object.
...	Additional arguments to be passed to other methods.
random	Distribution to describe random patterns.
aggregated	Distribution to describe aggregated patterns.
n_est	Number of estimated parameters for both distributions.

Details

Under the hood, distr_fit relies on the smle utility which is a wrapped around the optim procedure.

Note that there may appear warnings about chi-squared goodness-of-fit tests if any expected count is less than 5 (Cochran's rule of thumb).

Value

An object of class fit_two_distr, which is a list containing at least the following components:

call	The function call.
name	The names of both distributions.
model	The outputs of fitting process for both distributions.
llr	The result of the log-likelihood ratio test.

Other components can be present such as:

param	A numeric matrix of estimated parameters (that can be printed using printCoefmat).
freq	A data frame or a matrix with the observed and expected frequencies for both distributions for the different categories
gof	Goodness-of-fit tests for both distributions (which are typically chi-squared goodness-of-fit tests).

References

Madden LV, Hughes G. 1995. Plant disease incidence: Distributions, heterogeneity, and temporal analysis. Annual Review of Phytopathology 33(1): 529–564. doi:10.1146/annurev.py.33.090195.002525

Examples

```
# Simple workflow for incidence data:
my_data <- count(arthropods)
my_data <- split(my_data, by = "t")[[3]]
my_res  <- fit_two_distr(my_data)
summary(my_res)
plot(my_res)
```

```
# Simple workflow for incidence data:
my_data <- incidence(tobacco_viruses)
my_res  <- fit_two_distr(my_data)
summary(my_res)
plot(my_res)

# Note that there are other methods to fit some common distributions.
# For example for the Poisson distribution, one can use glm:
my_arthropods <- arthropods[arthropods$t == 3, ]
my_model <- glm(my_arthropods$i ~ 1, family = poisson)
lambda <- exp(coef(my_model)[[1]]) # unique(my_model$fitted.values) works also.
lambda
# ... or the fitdistr function in MASS package:
require(MASS)
fitdistr(my_arthropods$i, "poisson")

# For the binomial distribution, glm still works:
my_model <- with(tobacco_viruses, glm(i/n ~ 1, family = binomial, weights = n))
prob <- logit(coef(my_model)[[1]], rev = TRUE)
prob
# ... but the binomial distribution is not yet recognized by MASS::fitdistr.

# Examples featured in Madden et al. (2007).
# p. 242-243
my_data <- incidence(dogwood_anthracnose)
my_data <- split(my_data, by = "t")
my_fit_two_distr <- lapply(my_data, fit_two_distr)
lapply(my_fit_two_distr, function(x) x$param$aggregated[c("prob", "theta"), ])
lapply(my_fit_two_distr, plot)

my_agg_index <- lapply(my_data, agg_index)
lapply(my_agg_index, function(x) x$index)
lapply(my_agg_index, chisq.test)
```

`hop_viruses` *Incidence of three viruses in an Australian hop garden.*

Description

Three viruses, i.e. Hop latent virus (HpLV), Hop mosaic virus (HpMV), and Apple mosaic virus (ApMV), were monitored in an Australian hop garden for two consecutive years (1996 and 1997). The hop garden was established in 1989 with the variety Victoria in a commercial hop farm at Bushy Park, Tasmania, Australia. It consisted of 25 rows containing 51 plants each, so that there were 1275 hop plants in total. There were 2.1 m between rows, and 1.8 m between plants within rows.

Usage

```
hop_viruses
```

Format

There are three data frames, one for each virus (HpLV, HpMV and ApMV). Each data frame consists of 2550 rows and 7 variables:

[, 1:2]	x,y	Grid spatial coordinates.
[, 3:4]	xm,ym	Metric spatial coordinates.
[, 5]	t	Year of disease assessments.
[, 6]	i	Incidence. 0: Healthy, 1: Diseased.
[, 7]	n	Sampling unit size. n = 1 means that the sampling unit size is the plant.

Source

Pethybridge SJ, Madden LV. 2003. Analysis of spatiotemporal dynamics of virus spread in an Australian hop garden by stochastic modeling. Plant Disease 87:56-62. doi:10.1094/PDIS.2003.87.1.56

---

intensity *Construct count, incidence and severity objects.*

Description

count(), incidence() and severity() create eponym objects. All of these classes inherit from the base class intensity. The choice of the class depends on the nature of the data set.

Usage

```
count(data, mapping, keep_only_std = TRUE)

incidence(data, mapping, keep_only_std = TRUE)

severity(data, mapping, keep_only_std = TRUE)
```

Arguments

data	A data frame. Each line corresponds to a record (or case, or entry).
mapping	A mapping object, created with mapping() or mapping_() functions. ... A vector with all the corresponding variables. The different elements can be named (names of the elements) of the data frame in the incidence object), or unamed. In the latter case, elements must be correctly ordered, i.e. x, y, z, t, r and then n. If variables in NULL, then only the 6 first ... will be take into account in the following (1, 2, ...), i.e. the id of the value. All the 'parameters' need to be specified.
keep_only_std	Are only standard names kept when proceeding to mapping? Setting keep_only_std to TRUE may be useful for subsequent data splitting using extra labels.

Details

incidence reads disease incidence data from a data frame and return an incidence object. All of these classes inherit from intensity class.

- count: Each sampling unit contains from 0 to theoreticaly an infinity of data. Number are positive integers.
- incidence: Each sampling unit contains an number of diseased plants, ranging from 0 to n which is the total amount of plants per sampling unit.
- severity: Each sampling unit contain a percentage of disease, a positive real number ranging from 0.0 to 1.0.

Class intensity and inherited classes

All the classes recording disease intensity measurements inherit from this class. The class intensity is virtual which means that no object of a class intensity can be constructed. This class only describes common features of all the different disease intensity measurements implemented in this package (count, incidence and severity). You should call one of these inherited classes instead, depending on the nature of your data.

By convention, the first columns of the different data frames of each slots have names, but the spatial, temporal or even disease information do not need to fit to these conventions or may be less straightforward and need more columns to record correctly all the information. In such unusual situations, the automatic options of the analysis tools would need to be overridden to be able to work in the desired way.

The differences between the different inherited classes regard only the obs slot. In the case of count, the data expected for each record are positive integers (N+). For incidence, the data sets are supposed to be two information set per records, the number of diseased unit per sampling unit (r) and the total number of units per sampling unit (n). Note that in its current implementation, n is supposed to be the same for a whole data set. Unequal sampling units are not implemented yet. Finally, for severity, r is positive real ranging from 0 to 1 and depecting a percentage.

space A data frame containing only spatial information. Each row corresponds to a sampling unit. By convention, the first 3 columns are names x, y, z.

time A data frame containing temporal information. By convention, the first column is named t.

obs A data frame containing disease observations themselves. The name of the columns may differ between the sub-class chosed to record the data.

Note that it is possible to create a "severity" object but no statistical tools are currently implemented to deal with such an object.

An intensity object contains at very least the "pure" intensity records (column r) which is a so-called observational variable. Another observational variable, the number of individuals in a sampling unit (n), is present in the case of a incidence object. Very often in addition to observational variables, there are spatial (columns x, y and/or z) and/or temporal (column t) variables.

Note that the severity class and the z variable (the 3rd spatial dimension) are implemented but no statistical methods use them at this point.

Value

An intensity object.

When printed, difference information are available:

- The number of sampling units.
- The time.
- Is it georeferenced (TRUE/FALSE)
- Are there any NA data (TRUE/FALSE)
- Is it a complet array (TRUE/FALSE)? A complete array means that all the recorded values allow to display an array (even if some data are not available), but this was explicitelly specified. To complete a dataset, just use `complete(data)`. You can also remove NA, which is necessary to use some analysis technics, using `replaceNA(data)` or `replace.na(data)`. Note that using both commands will results in modifying the original data sets which will be specified.

Examples

```
## Create intensity objects
# Implicite call: The variable mapping does not need to be specified if the
# column names of the input data frame follow the default names.
colnames(tomato_tswv$field_1929) # Returns c("x", "y", "t", "i", "n")
my_incidence_1 <- incidence(tomato_tswv$field_1929)
my_incidence_1
my_incidence_2 <- incidence(tomato_tswv$field_1929,
                            mapping(x = x, y = y, t = t, i = i, n = n))
identical(my_incidence_1, my_incidence_2)

# Explicite call: Otherwise, the variable mapping need to be specified, at
# least for column names that do not correspond to default names.
colnames(aphids) # Returns c("xm", "ym", "i")
my_count_1 <- count(aphids, mapping(x = xm, y = ym, i = i))
my_count_1
# We can drop the "i = i" in the mapping.
my_count_2 <- count(aphids, mapping(x = xm, y = ym))
identical(my_count_1, my_count_2)

# It is possible to change the variable mapping after the creation of an
# intensity object:
another_incidence <- incidence(hop_viruses$HpLV)
another_incidence
remap(another_incidence, mapping(x = xm, y = ym))

## Plotting data
plot(my_incidence_1) # Same as: plot(my_incidence_1, type = "spatial")
plot(my_incidence_1, type = "temporal")

plot(my_count_1, tile = FALSE, size = 5)
plot(my_count_1, type = "temporal") # Not possible: there is only 1 date.

# Using grayscale:
plot(my_count_1, grayscale = TRUE)
plot(my_count_1, grayscale = TRUE, tile = FALSE, size = 5)
```

is.intensity *Test if an object is of class* intensity *or one of its subclasses.*

Description

Test if an object is of class intensity or one of its subclasses (i.e. count, incidence or severity).

Usage

```
is.intensity(x)

is.count(x)

is.incidence(x)

is.severity(x)
```

Arguments

x	An object.

Value

TRUE if its argument's value is the corresponding intensity object and FALSE otherwise.

link *Some link functions.*

Description

Logit, probit and cloglog functions are available. The logit and the logistic (with rev = TRUE), i.e. the inverse-logit functions. Probit is a wrapper around qnorm (for $probit$) and pnorm (for $probit^{-1}$) Complementary log-log transformation.

Usage

```
logit(x, rev = FALSE)

probit(x, rev = FALSE)

cloglog(x, rev = FALSE)
```

Arguments

x	A numeric vector.
rev	The inverse of the function?

Value

A numeric vector.

---

mapcomp *Map Comparison procedure.*

Description

mapcomp performs a spatial pattern analysis based on the calculation of a formal distance (the Hellinger distance) between the density map of count or incidence data, and the density map of sampling effort. Statistical tests of spatial homogeneity are based on permutations across sampling sites and on valuable properties of the Hellinger distance.

Usage

```
mapcomp(data, ...)

## S3 method for class 'data.frame'
mapcomp(
  data,
  delta,
  bandwidth,
  nperm = 100,
  edge_correction = FALSE,
  threads = 1,
  verbose = TRUE,
  ...
)

## S3 method for class 'matrix'
mapcomp(
  data,
  delta,
  bandwidth,
  nperm = 100,
  edge_correction = FALSE,
  threads = 1,
  verbose = TRUE,
  ...
)

## S3 method for class 'count'
mapcomp(
  data,
  delta,
  bandwidth,
```

```
  nperm = 100,
  edge_correction = FALSE,
  threads = 1,
  verbose = TRUE,
  ...
)

## S3 method for class 'incidence'
mapcomp(
  data,
  delta,
  bandwidth,
  nperm = 100,
  edge_correction = FALSE,
  threads = 1,
  verbose = TRUE,
  ...
)
```

Arguments

data	A data frame or a matrix with only three columns: the two first ones must be the x and y coordinates of the sampling units, and the last one, the corresponding disease intensity observations. It can also be a count or an incidence object.
...	Additional arguments to be passed to other methods.
delta	Mesh size of the grid over the geographical domain of the sampling units used to compute the integral Hellinger distance between the probability density function of observations and the probability density function of sampling effort.
bandwidth	Bandwidth parameter for smoothing. It allows to test the spatial extent of heterogeneity if any.
nperm	Number of random permutations to assess probabilities.
edge_correction	Apply edge correction to account for the fact that bordering points intrinsically suffer from a lack of neighboring observation sites. FALSE by default.
threads	Number of threads to perform the computations.
verbose	Explain what is being done (TRUE by default).

Value

An object of class mapcomp, which is a list containing the following components:

data	The input data.
coord	The coordinates and normalized intensity for each point of the full grid.
object	The class of data.
bandwidth	The bandwidth parameter.
stat, pval	The statistic and corresponding p-value (see references for more details).

References

Lavigne C, Ricci B, Franck P, Senoussi R. 2010. Spatial analyses of ecological count data: A density map comparison approach. Basic and Applied Ecology. 11:734–742.

Examples

```
set.seed(123)
my_res <- mapcomp(codling_moths, delta = 1, bandwidth = 11,
                  edge_correction = FALSE, nperm = 20)
my_res
plot(my_res)

set.seed(123)
my_count <- count(codling_moths, mapping(x = xm, y = ym))
my_res <- mapcomp(my_count, delta = 1, bandwidth = 11,
                  edge_correction = FALSE, nperm = 20)
my_res
plot(my_res, bins = 10)
```

---

mapped_var *Existing variable mappings.*

Description

Get or set existing variable mappings.

Usage

```
mapped_var(x)

mapped_var(x, keep = TRUE) <- value
```

Arguments

x	An intensity object.
keep	Logical. Do we keep any previous mapped variables that are not redifined in the mapping object?
value	A mapping object.

Value

mapped_var returns the list of current mapped names of the object x.

See Also

mapping

Examples

```
my_data <- count(aphids)
my_data
mapped_var(my_data)
mapped_var(my_data) <- mapping(x = X, y = Y)
mapped_var(my_data)
mapped_var(my_data) <- mapping(x = x, r = r, keep = FALSE)
mapped_var(my_data)
```

`mapping` *Construct data mappings.*

Description

Data mappings describe how variables in the data are mapped to standard names used throughout `epiphy`.

Usage

```
mapping(...)

mapping_(x)

remap(data, mapping, keep_only_std = TRUE)
```

Arguments

`...`	One or more unquoted expressions separated by commas.
`x`	Vector of one or more character strings.
`data`	An `intensity` object.
`mapping`	A `mapping` object.
`keep_only_std`	Keep only standard variables.

Details

Standard names are `x`, `y` and `z` for the three spatial dimensions, and `t` for the time. `r` corresponds to the records of (disease) intensity, and `n`, the number of individuals in a sampling unit (if applicable).

`mapping()` works with expressions, and `mapping_()`, with a vector of characters.

Value

A list of mapped names.

See Also

`mapped_var`

Examples

```
mapping(x = col1, y = col2)
mapping_(c("x = col1", "y = col2"))
```

`offspring_survival` *Offspring survival of rats experiencing different diets.*

Description

Results of an experiment where two groups of 16 female rats were fed different diets during pregnancy and lactation periods. One group's diet contained a chemical under review, and the other one was a control. For each litter, the number of pups alive at 4 days, and the number of pups weaned (i.e. that survived the 21-day lactation period) were recorded.

Usage

```
offspring_survival
```

Format

A data frame with 32 rows and 3 variables:

[, 1]	group	Either control or treated group.
[, 2]	i	Pups weaned.
[, 3]	n	Pups alive at 4 days.

Source

Weil CS. 1970. Selection of the valid number of sampling units and a consideration of their combination in toxicological studies involving reproduction, teratogenesis or carcinogenesis. Food and Cosmetics Toxicology 8: 177-182.

`onion_bacterial_blight`

Incidence of bacterial blight of onion.

Description

Assessments of bacterial blight of onion at two dates. The experimental plot was sown with naturally X. axonopodis pv. allii-contaminated onion (A. cepa L. cv. Chateau-vieux) seed lot, with a contamination rate of about 0.04%.

Usage

```
onion_bacterial_blight
```

Format

A data frame with 1134 rows and 5 variables:

[, 1:2]	x,y	Grid spatial coordinates.
[, 3]	t	Date of disease assessments.
[, 4]	i	Disease incidence. 0: Healthy, 1: Diseased.
[, 5]	n	Sampling unit size. n = 1 means that the sampling unit size is the plant.

Source

Roumagnac P, Pruvost O, Chiroleu F, Hughes G. 2004. Spatial and temporal analyses of bacterial blight of onion caused by Xanthomonas axonopodis pv. allii. Phytopathology 94, 138–146. doi:10.1094/PHYTO.2004.94.2.138

---

power_law *Taylor's and binary power laws.*

Description

Assesses the overall degree of heterogeneity in a collection of data sets at the sampling-unit scale.

Usage

```
power_law(data, log_base = exp(1), ...)
```

Arguments

data	A list of intensity objects (count or incidence objects).
log_base	Logarithm base to be used.
...	Additional arguments to be passed to other methods.

Details

The power law describes the relationship between the observed variance of individuals within a data set (s^2) and the corresponding variance under the assumption of no aggregation (s\'^2). It can be expressed under its logarithmic form as: log(s^2) = log(a) + b log(Y), with:

- Y = p in the case of count data (Taylor's power law).
- Y = p(1 - p) in the case of incidence data (binary power law).

p corresponds to the mean proportion of recorded individuals in case of incidence data, and the absolute value in case of count data.

Value

A power_law object.

References

Taylor LR. 1961. Aggregation, variance and the mean. Nature 189: 732–35.

Hughes G, Madden LV. 1992. Aggregation and incidence of disease. Plant Pathology 41 (6): 657–660. doi:10.1111/j.13653059.1992.tb02549.x

Madden LV, Hughes G, van den Bosch F. 2007. Spatial aspects of epidemics - III: Patterns of plant disease. In: The study of plant disease epidemics, 235–278. American Phytopathological Society, St Paul, MN.

Examples

```
require(magrittr)
my_data <- do.call(c, lapply(citrus_ctv, function(citrus_field) {
   incidence(citrus_field) %>%
       clump(unit_size = c(x = 3, y = 3)) %>%
       split(by = "t")
}))
# my_data is a list of incidence object, each one corresponding to a given
# time at a given location.
my_power_law <- power_law(my_data)
my_power_law
summary(my_power_law)
plot(my_power_law) # Same as: plot(my_power_law, scale = "log")
plot(my_power_law, scale = "lin")
```

---

pyrethrum_ray_blight *Incidence of ray blight disease of pyrethrum.*

Description

An assessment of the incidence of ray blight disease of pyrethrum in 62 sampling units, containing 6 plants each.

Usage

```
pyrethrum_ray_blight
```

Format

A data frame with 62 rows and 2 variables:

[, 1]	i	Number of diseased plants (from 0 to 6).
[, 2]	n	Sampling unit size. Here, n = 6 plants per sampling unit.

Source

Pethybridge SJ, Esker P, Hay F, Wilson C, Nutter FW. 2005. Spatiotemporal description of epidemics caused by Phoma ligulicola in Tasmanian pyrethrum fields. *Phytopathology* 95, 648–658. doi:10.1094/PHYTO950648

`sadie` *Spatial Analysis by Distance IndicEs (SADIE).*

Description

`sadie` performs the SADIE procedure. It computes different indices and probabilities based on the distance to regularity for the observed spatial pattern and a specified number of random permutations of this pattern. Both kind of clustering indices described by Perry et al. (1999) and Li et al. (2012) can be computed.

Usage

```
sadie(data, ...)

## S3 method for class 'data.frame'
sadie(
  data,
  index = c("Perry", "Li-Madden-Xu", "all"),
  nperm = 100,
  seed = NULL,
  threads = 1,
  ...,
  method = "shortsimplex",
  verbose = TRUE
)

## S3 method for class 'matrix'
sadie(
  data,
  index = c("Perry", "Li-Madden-Xu", "all"),
  nperm = 100,
  seed = NULL,
  threads = 1,
  ...,
  method = "shortsimplex",
  verbose = TRUE
)

## S3 method for class 'count'
```

```
sadie(
  data,
  index = c("Perry", "Li-Madden-Xu", "all"),
  nperm = 100,
  seed = NULL,
  threads = 1,
  ...,
  method = "shortsimplex",
  verbose = TRUE
)

## S3 method for class 'incidence'
sadie(
  data,
  index = c("Perry", "Li-Madden-Xu", "all"),
  nperm = 100,
  seed = NULL,
  threads = 1,
  ...,
  method = "shortsimplex",
  verbose = TRUE
)
```

Arguments

data	A data frame or a matrix with only three columns: the two first ones must be the x and y coordinates of the sampling units, and the last one, the corresponding disease intensity observations. It can also be a count or an incidence object.
...	Additional arguments to be passed to other methods.
index	The index to be calculated: "Perry", "Li-Madden-Xu" or "all". By default, only Perry's index is computed for each sampling unit.
nperm	Number of random permutations to assess probabilities.
seed	Fixed seed to be used for randomizations (only useful for checking purposes). Not fixed by default (= NULL).
threads	Number of threads to perform the computations.
method	Method for the transportation algorithm.
verbose	Explain what is being done (TRUE by default).

Details

By convention in the SADIE procedure, clustering indices for a donor unit (outflow) and a receiver unit (inflow) are positive and negative in sign, respectively.

Value

A sadie object.

References

Perry JN. 1995. Spatial analysis by distance indices. Journal of Animal Ecology 64, 303–314. doi:10.2307/5892

Perry JN, Winder L, Holland JM, Alston RD. 1999. Red–blue plots for detecting clusters in count data. Ecology Letters 2, 106–113. doi:10.1046/j.14610248.1999.22057.x

Li B, Madden LV, Xu X. 2012. Spatial analysis by distance indices: an alternative local clustering index for studying spatial patterns. Methods in Ecology and Evolution 3, 368–377. doi:10.1111/j.2041210X.2011.00165.x

Examples

```
set.seed(123)
# Create an intensity object:
my_count <- count(aphids, mapping(x = xm, y = ym))
# Only compute Perry's indices:
my_res <- sadie(my_count)
my_res
summary(my_res)
plot(my_res)
plot(my_res, isoclines = TRUE)

set.seed(123)
# Compute both Perry's and Li-Madden-Xu's indices (using multithreading):
my_res <- sadie(my_count, index = "all", threads = 2, nperm = 20)
my_res
summary(my_res)
plot(my_res) # Identical to: plot(my_res, index = "Perry")
plot(my_res, index = "Li-Madden-Xu")

set.seed(123)
# Using usual data frames instead of intensity objects:
my_df <- aphids[, c("xm", "ym", "i")]
sadie(my_df)
```

---

simulated_epidemics *Examples of simulated epidemic data.*

Description

Epidemics were generated using the stochastic simulator from Xu and Madden (2004). The data consist of the numbers of diseased plants per sampling unit (out of a total of n = 100 plants in each sampling unit). N = 144 sampling units, and different values for the parameters `pattern` and `mu` were used for the simulations.

Usage

```
simulated_epidemics
```

Format

A data frame with 864 rows and 6 variables:

[, 1]	pattern	Either clumped (i.e. aggregated), random or regular.
[, 2]	mu	Median spore dispersal parameter.
[, 3:4]	x,y	Grid spatial coordinates.
[, 5]	i	Number of diseased plants (from 0 to 100).
[, 6]	n	Sampling unit size. Here, n = 100 plants per sampling unit.

Source

Xu XM, Madden LV. 2004. Use of SADIE statistics to study spatial dynamics of plant disease epidemics. Plant Pathology 53, 38–49. doi:10.1111/j.13653059.2004.00949.x

---

`spatial_hier` *Spatial hierarchy analysis.*

Description

The manner in which the data are collected provides information about aggregation of disease at different levels in a spatial hierarchy (Hughes et al. 1997). For example, a sampling unit (upper level) can be reported as "healthy", if no diseased leaves (lower level) were found within the sampling unit.

Usage

```
spatial_hier(low, high)
```

Arguments

`low`	An list of `intensity` objects.
`high`	An list of `intensity` objects.

Details

In a pairwise comparison between levels, the probability that an individual at the lower hierarchical level is diseased is denoted plow, and phigh refers to the probability of disease at the higher level. The relationship between these two probabilities can be written as

phigh = 1 - (1 - plow)^nu

where n is a parameter ranging from 0 to the corresponding number of individuals at the hierarchical level referenced by plow. If the value of n is equal to the number of individuals at the lower hierarchical level contained in a unit of the higher level (n low), this suggests that there is no aggregation of disease incidence at the lower level. Conversely, a value of n less than n low is indicative of aggregation at that level. The value of n can be interpreted as an effective sample size

(Hughes and Gottwald 1999; Madden and Hughes 1999) in the statistical sense that its value indicates the number of independent pieces of information at the lower level. Here, the effective sample size concerns the equating of the zero-term of the binomial distribution with the zero-term of an overdispersed distribution, as described in Madden and Hughes (1999). Using the complementary log-log transformation, CLL(x) = ln(-ln(1-x)), one can rewrite the Equation 5 as follows (Madden et al. 2007):

CLL(phigh) = ln(nu) + CLL(plow)

from which the value of ln(n) can be obtained as the intercept of a linear regression when the slope is constrained to 1.

Value

A `spatial_hier` object.

Examples

```
my_data_low <- incidence(tomato_tswv$field_1929)
my_data_low <- clump(my_data_low, c(x = 3, y = 3))
my_data_high <- my_data_low
my_data_high$data$n <- 1
my_data_high$data$i <- ifelse(my_data_high$data$i > 0, 1, 0)
my_data_low  <- split(my_data_low, by = "t")
my_data_high <- split(my_data_high, by = "t")
res <- spatial_hier(my_data_low, my_data_high)

res
summary(res)
plot(res)
```

`split.intensity` *Divide into groups and reassemble.*

Description

Divide into groups and reassemble.

Usage

```
## S3 method for class 'intensity'
split(x, f, drop = FALSE, ..., by, unit_size)
```

Arguments

`x`	vector or data frame containing values to be divided into groups.

f	a 'factor' in the sense that as.factor(f) defines the grouping, or a list of such factors in which case their interaction is used for the grouping. If x is a data frame, f can also be a formula of the form ~ g to split by the variable g, or more generally of the form ~ g1 + ... + gk to split by the interaction of the variables g1, ..., gk, where these variables are evaluated in the data frame x using the usual non-standard evaluation rules.
drop	logical indicating if levels that do not occur should be dropped (if f is a factor or a list).
...	further potential arguments passed to methods.
by	The name(s) of the variable(s) which define(s) the grouping.
unit_size	Size of a group unit. It must be a named vector, with names corresponding to non-observational variables (i.e. space and time variables). If the size of a variable in the data set is not a multiple of the provided value in unit_size, some sampling units (the last ones) will be dropped so that clumps of individuals remain even throughout the data set.

Value

A list of intensity objects.

Examples

```
my_incidence <- incidence(tomato_tswv$field_1929)
plot(my_incidence, type = "all")
my_incidence_spl1 <- split(my_incidence, by = "t")
my_incidence_spl2 <- split(my_incidence, unit_size = c(x = 8, y = 20, t = 1))
```

---

threshold *To go to higher level in the hierarchy.*

Description

This function transforms the current numeric vector or intensity data set into a "simplified black and white image" of this same data set: every value of disease intensity below and above a given threshold is given the value 0 and 1, respectively.

Usage

```
threshold(data, value, ...)
```

Arguments

data	A numeric vector or an intensity object.
value	All the intensity values lower or equal to this value are set to 0. The other values are set to 1.
...	Additional arguments to be passed to other methods.

Details

By default, everything above 0 is given 1, and 0 stays at 0. `threshold` is thus useful to report a whole sampling unit as "healthy" (0), if no diseased individual at all was found within the sampling unit, or "diseased" (1) if at least one diseased individual was found.

Value

A numeric vector or an `intensity` object.

Examples

```
my_incidence <- incidence(tomato_tswv$field_1929)
plot(my_incidence, type = "all")
my_incidence_clumped_1 <- clump(my_incidence, unit_size = c(x = 3, y = 3))
plot(my_incidence_clumped_1, type = "all")
my_incidence_thr <- threshold(my_incidence_clumped_1, value = 4)
plot(my_incidence_thr, type = "all")
```

---

tobacco_viruses *Incidence of tobacco plants infected with viruses.*

Description

Experimental plot consisted of 75 sampling units with 40 tobacco plants in each one.

Usage

```
tobacco_viruses
```

Format

A data frame with 75 rows and 2 variables:

[, 1]	i	Number of diseased plants (from 0 to 40).
[, 2]	n	Sampling unit size. Here, n = 40 plants per sampling unit.

Source

Madden LV, Pirone TP, Raccah B. 1987. Analysis of spatial patterns of virus-diseased tobacco plants. Phytopathology 77, 1409–1417.

`tomato_tswv` *Incidence of tomato spotted wilt virus (TSWV) disease in field trials.*

Description

Intensively mapped TSWV incidence data reported by Cochran (1936) and Bald (1937). The disease assessments were performed in field trials at the Waite Institute (Australia) in 1928 and 1929. TSWV is a virus disease spread by thrips.

Usage

```
tomato_tswv
```

Format

There are two data frames:

`field_1928`: A data frame with 11088 rows and 8 variables:

[, 1]	plot	Plot id.
[, 2]	variety	Variety name.
[, 3]	irrigation	Irrigation system.
[, 4:5]	x,y	Grid spatial coordinates.
[, 6]	t	Date of disease assessments. 1: 6 Nov, 2: 14 Nov, 3: 21 Nov, 4: 28-29 Nov, 5: 5 Dec, 6: 12 Dec 1928.
[, 7]	i	Disease incidence. 0: Healthy, 1: Diseased.
[, 8]	n	Sampling unit size. n = 1 means that the sampling unit size is the plant.

`field_1929`: A data frame with 4320 rows and 5 variables:

[, 1:2]	x,y	Grid spatial coordinates.
[, 3]	t	Date of disease assessments. 1: 18 Dec, 2: 31 Dec 1929, 3: 22 Jan 1930.
[, 4]	i	Disease incidence. 0: Healthy, 1: Diseased.
[, 5]	n	Sampling unit size. n = 1 means that the sampling unit size is the plant.

Details

The data set `field_1928`, reported by Bald (1937), was a set of four plots. Each plot consisted of 14 rows containing 33 plants each, so that there were 462 plants in each plot. The tomato variety Early Dwarf Red was used in two plots, and the variety Burwood Prize in the other two. The tomatoes were planted out on 15th October 1928. The two plots dedicated to a given variety experienced different irrigation practices, using either overhead sprays or trenches. Otherwise, all were treated alike. Weekly records of TSWV incidence were performed from 6th November to 12th December.

The data set `field_1929`, reported by Cochran (1936), was a field of 24 rows containing 60 plants each, so that there were 1440 plants. The tomatoes were planted out in 26th November 1929. TSWV incidence records made on 18th December 1929, 31st December 1929 and 22nd January 1930 are reported in this data set.

Source

Cochran WG. 1936. The statistical analysis of field counts of diseased plants. Supplement to the Journal of the Royal Statistical Society 3, 49–67. doi:10.2307/2983677

Bald JG. 1937. Investigations on "spotted wilt" of tomatoes. III. Infection in field plots. Bulletin 106. Melbourne, Australia: Council for Scientific and Industrial Research.

`z.test` *Z-test.*

Description

Performs z-tests for Fisher's aggregation indices (computed with either count or incidence data).

Usage

```
z.test(x, ...)

## Default S3 method:
z.test(x, ...)

## S3 method for class 'fisher'
z.test(
  x,
  alternative = c("two.sided", "less", "greater"),
  conf.level = 0.95,
  ...
)
```

Arguments

`x`	The output of the `agg_index` function with `method = "fisher"` as parameter.
`...`	Not yet implemented.
`alternative`	A character string specifying the alternative hypothesis. It must be one of "two.sided" (default), "less" or "greater".
`conf.level`	The confidence level of the interval.

Details

For two-sided tests with a confidence level of 95 the spatial pattern would be random. If z < -1.96 or z > 1.96, it would be uniform or aggregated, respectively.

Value

Same kind of object as the one returns by the stats `chisq.test` function for example.

References

For count and incidence data:

Moradi-Vajargah M, Golizadeh A, Rafiee-Dastjerdi H, Zalucki MP, Hassanpour M, Naseri B. 2011. Population density and spatial distribution pattern of Hypera postica (Coleoptera: Curculionidae) in Ardabil, Iran. Notulae Botanicae Horti Agrobotanici Cluj-Napoca, 39(2): 42-48.

Sun P, Madden LV. 1997. Using a normal approximation to test for the binomial distribution. Biometrical journal, 39(5): 533-544.

See Also

calpha.test, chisq.test

Examples

```
# For incidence data:
my_incidence <- incidence(tobacco_viruses)
my_fisher <- agg_index(my_incidence, method = "fisher")
z.test(my_fisher)
```

Index