Package 'multiflexscan'

October 26, 2026

Type Package

Title Information Criterion and Scan Statistic Approach for Detecting Multiple Disease Clusters

Version 0.3.0

Author Takahiro Otani [aut, cre] (ORCID: <https://orcid.org/0000-0002-4007-7817>), Kunihiko Takahashi [aut]

Maintainer Takahiro Otani <t.otani@aichi-cc.jp>

Description Detecting multiple disease clusters using the information criterion and scan statistic approach developed by Takahashi and Shimadzu (2020) <doi:10.1186/s12942-020-00228-y>.

License GPL-3

URL https://github.com/tkhrotn/multiflexscan

BugReports https://github.com/tkhrotn/multiflexscan/issues

Depends R (>= 4.0.0)

Imports doRNG, doSNOW, foreach, igraph, parallel, rflexscan (>= 1.2.0), utils

Suggests knitr, rmarkdown, sf, spdep, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

Config/roxygen2/version 8.1.0

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2026-10-26 05:10:02 UTC

Contents

multiflexscan-package . . . 2
AIC.multiflexscan . . . 3
as.data.frame.multiflexscan . . . 4
choropleth . . . 5
clusters . . . 6
coef.multiflexscan . . . 7
get_setting . . . 8
logLik.multiflexscan . . . 9
model_criteria . . . 10
multiflexscan . . . 11
nclusters . . . 14
nobs.multiflexscan . . . 15
plot.multiflexscan . . . 16
print.multiflexscan . . . 17
print.summary.multiflexscan . . . 18
pvalue . . . 19
summary.multiflexscan . . . 20

Index **21**

`multiflexscan-package` *Detecting multiple spatial disease clusters using the information criterion and scan statistic approach*

Description

This package provides functions for detecting multiple spatial disease clusters based on the information criterion and scan statistic approach by Takahashi and Shimadzu (2020), using the flexible spatial scan statistic (Tango and Takahashi, 2005) or Kulldorff's circular spatial scan statistic (1997) implemented in the rflexscan package (Otani and Takahashi, 2021).

Author(s)

Maintainer: Takahiro Otani <t.otani@aichi-cc.jp> (ORCID)

Authors:

- Takahiro Otani <t.otani@aichi-cc.jp> (ORCID)
- Kunihiko Takahashi

References

- Takahashi K. and Shimadzu H. (2020). Detecting multiple spatial disease clusters: information criterion and scan statistic approach. International Journal of Health Geographics 19:33. doi:10.1186/s1294202000228y
- Tango T. and Takahashi K. (2005). A flexibly shaped spatial scan statistic for detecting clusters, International Journal of Health Geographics 4:11.

- Otani T. and Takahashi K. (2021). Flexible scan statistics for detecting spatial disease clusters: The rflexscan R package. Journal of Statistical Software 99:13. doi:10.18637/jss.v099.i13
- Takahashi K, Yokoyama T and Tango T. (2010). FleXScan v3.1: Software for the Flexible Scan Statistic. National Institute of Public Health, Japan, https://sites.google.com/site/flexscansoftware/home.

See Also

Useful links:

- https://github.com/tkhrotn/multiflexscan
- Report bugs at https://github.com/tkhrotn/multiflexscan/issues

---

`AIC.multiflexscan` *AIC and BIC for the selected cluster model*

Description

AIC and BIC for the selected cluster model

Usage

```
## S3 method for class 'multiflexscan'
AIC(object, ..., k = 2)

## S3 method for class 'multiflexscan'
BIC(object, ...)
```

Arguments

`object`	A `multiflexscan` object.
`...`	Optionally, additional fitted model objects for comparison.
`k`	Numeric penalty per estimated parameter for `AIC`. The default, 2, gives ordinary AIC.

Details

These methods delegate to the standard R methods via `logLik()`. AIC and BIC are distinct from the criterion used to select the clusters; use `C_criterion()` or `RDC()` for that criterion. Comparisons require models fitted to the same response data with comparable likelihoods.

Value

With one object, the AIC or BIC of the selected K-cluster model. With multiple objects, a data frame containing `df` and `AIC` or `BIC`, as in `stats::AIC()`.

See Also

model_criteria(), logLik.multiflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
AIC(fit)
BIC(fit)
AIC(fit, k = log(nobs(fit)))
```

---

as.data.frame.multiflexscan

Coerce selected clusters to a data frame

Description

Coerce selected clusters to a data frame

Usage

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

Arguments

x	A multiflexscan object.
row.names	Optional row names (ignored).
optional	Logical; ignored.
...	Additional arguments (ignored).

Value

A data frame of selected clusters (see clusters()).

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
as.data.frame(fit)
```

---

choropleth *Draw a choropleth map of multiflexscan clusters*

Description

Draw a choropleth map that highlights clusters selected by the information criterion. The order of `regions` must match the input order used in `multiflexscan()`.

Usage

```
choropleth(
  x,
  regions,
  selected = seq_len(nclusters(x)),
  col = grDevices::palette(),
  background = "grey95",
  border = "grey80",
  border.lwd = 0.25,
  ...
)
```

Arguments

`x`	A `multiflexscan` object.
`regions`	An `sf` object containing the region geometries.
`selected`	Integer vector specifying the clusters to highlight. By default, the clusters selected by the information criterion are highlighted.
`col`	Colors used for highlighted clusters.
`background`	Fill color for regions not in selected clusters.
`border`	Border color passed to the geometry plot.
`border.lwd`	Line width for polygon borders. A value smaller than the graphics default is used by default to reduce overlap in dense maps.
`...`	Additional arguments passed to the geometry plot.

Value

The `regions` object with an added `cluster` column (integer cluster membership, `0` for unclustered regions), invisibly.

Note

Requires the suggested **sf** package.

See Also

multiflexscan, plot.multiflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
if (requireNamespace("sf", quietly = TRUE)) {
  polygons <- lapply(seq_len(nrow(grid)), function(i) {
    x <- grid$x[i]; y <- grid$y[i]
    sf::st_polygon(list(cbind(x + c(-.5, .5, .5, -.5, -.5),
                              y + c(-.5, -.5, .5, .5, -.5))))
  })
  regions <- sf::st_sf(id = seq_len(25), geometry = sf::st_sfc(polygons))
  choropleth(fit, regions)
}
```

---

clusters *Cluster summary table*

Description

Return a data frame of candidate clusters. By default only the clusters selected by the information criterion are returned.

Usage

```
clusters(object, selected = TRUE)
```

Arguments

object	A multiflexscan object.
selected	Logical; if TRUE (default), return only the selected clusters. If FALSE, return all candidate clusters.

Value

A data frame with columns NumArea, MaxDist, Case, Expected, RR, Stats, and P.

See Also

nclusters(), as.data.frame.multiflexscan, coef.multiflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
clusters(fit)
clusters(fit, selected = FALSE)
```

---

coef.multiflexscan *Extract selected cluster summaries*

Description

Extract selected cluster summaries

Usage

```
## S3 method for class 'multiflexscan'
coef(object, ...)
```

Arguments

object	A multiflexscan object.
...	Additional arguments (ignored).

Value

A data frame of selected clusters (see clusters()).

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
coef(fit)
```

---

get_setting *Analysis settings*

Description

Analysis settings

Usage

```
get_setting(object)
```

Arguments

object	A multiflexscan object.

Value

A list of analysis settings stored in the object.

See Also

nclusters(), pvalue()

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
get_setting(fit)
```

`logLik.multiflexscan` *Log-likelihood of the selected cluster model*

Description

Log-likelihood of the selected cluster model

Usage

```
## S3 method for class 'multiflexscan'
logLik(object, ...)
```

Arguments

`object`	A `multiflexscan` object.
`...`	Unused; additional arguments are not supported.

Details

The likelihood includes the Poisson normalizing constants and is conditional on the candidate regions. No model is refitted. The effective parameter count is recovered from the stored identity $AIC = -2 \log L + 2df$; it is not assumed to equal $K + 1$ when the model matrix is rank deficient. The Poisson dispersion is fixed.

Value

An object of class `logLik`, containing the selected Poisson model's log-likelihood, with attributes `df` (effective number of estimated parameters) and `nobs` (number of regions).

See Also

`model_criteria()`, `stats::AIC()`, `nobs()`

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
logLik(fit)
-2 * as.numeric(logLik(fit))
AIC(fit)
BIC(fit)
```

`model_criteria` *Extract model-selection criteria*

Description

Retrieve stored diagnostics without rerunning the analysis.

Usage

```
model_criteria(object, selected = FALSE)

C_criterion(object)

RDC(object)
```

Arguments

`object`	A `multiflexscan` object.
`selected`	Logical scalar. If `FALSE` (default), return all candidate models, including the null model with $K = 0$. If `TRUE`, return only the model selected by the cluster criterion.

Details

The selected model minimizes `C` (equivalently maximizes `RDC`); it does not necessarily minimize AIC or BIC. Ties are resolved in favor of the smallest K. The returned values describe Poisson models conditional on the candidate regions found by the scan.

Value

`model_criteria()` returns a data frame with columns `K`, `neg2logLik`, `AIC`, `BIC`, `C`, and `RDC`. `C_criterion()` and `RDC()` return the selected model's corresponding criterion as a numeric scalar.

See Also

`nclusters()`, logLik.multiflexscan, AIC.multiflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
```

```
  simcount = 19, cores = 1, seed = 20261002)
criteria <- model_criteria(fit)
model_criteria(fit, selected = TRUE)
plot(C ~ K, data = criteria, type = "b")
C_criterion(fit)
RDC(fit)
```

`multiflexscan` *Detect multiple spatial clusters using the flexible/circular scan statistic*

Description

This function analyzes spatial count data using the flexible spatial scan statistic developed by Tango and Takahashi (2005) or Kulldorff's circular spatial scan statistic (1997), and detects multiple disease clusters by selecting the number of clusters with an information criterion.

Usage

```
multiflexscan(
  x,
  y,
  name,
  observed,
  expected,
  nb,
  clustertype = "HOT",
  clustersize = 15,
  clusterradius = .Machine$double.xmax,
  maxclusters = 10,
  stattype = "ORIGINAL",
  scanmethod = "FLEXIBLE",
  ralpha = 0.2,
  simcount = 999,
  verbose = FALSE,
  cores = max(1, parallel::detectCores() - 1),
  parallel_type = "PSOCK",
  seed = NULL
)
```

Arguments

`x`	A vector of X-coordinates.
`y`	A vector of Y-coordinates.
`name`	A vector of names of each area.
`observed`	A vector with the observed number of disease cases.
`expected`	A vector with the expected number of disease cases under the null hypothesis. This is used with the Poisson model.

nb	A neighbours list or an adjacency matrix. When a list is supplied, element nb[[i]] gives the indices of regions adjacent to region i. When a matrix is supplied, non-zero entries indicate adjacency (the diagonal is ignored in the stored result).
clustertype	Type of cluster to be scanned. **"HOT"** Hot-spot clusters with elevated risk. **"COLD"** Cold-spot clusters with reduced risk. **"BOTH"** Hot- and cold-spot clusters simultaneously.
clustersize	Maximum spatial cluster size S, i.e., the maximum number of regions included in a detected cluster.
clusterradius	Upper bound on the spatial radius of candidate scanning windows, in the same units as x and y. The default .Machine$double.xmax imposes no radius limit. Choose a value consistent with the projected coordinate system used for x and y.
maxclusters	The maximum number of clusters to be detected.
stattype	Statistic type to be used (case-insensitive). **"ORIGINAL"** the likelihood ratio statistic by Kulldorff and Nagarwalla (1995) **"RESTRICTED"** the restricted likelihood ratio statistic by Tango (2008), with a preset parameter ralpha for restriction
scanmethod	Scanning method to be used (case-insensitive). **"FLEXIBLE"** flexible scan statistic by Tango and Takahashi (2005) **"CIRCULAR"** circular scan statistic by Kulldorff (1997)
ralpha	Threshold parameter of the mid-p-value for the restricted likelihood ratio statistic.
simcount	The number of Monte Carlo replications to calculate a p-value for statistical test.
verbose	Print progress messages.
cores	The number of CPU cores used for Monte Carlo replications.
parallel_type	Type of parallel backend passed to parallel::makeCluster() when distributing Monte Carlo replications across worker processes (default "PSOCK").
seed	Optional integer seed for reproducible Monte Carlo replications. When cores > 1, independent random streams are managed with **doRNG**; calling set.seed() alone is not sufficient for parallel reproducibility. The default NULL leaves the RNG unseeded.

Details

Centroid coordinates for each region should be specified by Cartesian coordinates using arguments x and y. We recommend transforming latitude and longitude onto an appropriate projected coordinate system before calling this function.

The function first obtains non-overlapping candidate clusters by calling rflexscan::runFleXScan(), then fits a sequence of Poisson generalized linear models with cluster indicators and the expected counts as offsets. The selected number of clusters is the value that maximizes the relative difference criterion (RDC). The global p-value is calculated by Monte Carlo simulation under the null hypothesis of no clustering.

The relative difference criterion for a model with K clusters is

$$\mathrm{RDC}_K = \frac{C_0 - C_K}{C_0},$$

where C_K is the information criterion of Takahashi and Shimadzu (2020). The selected number of clusters `nclust` is the value of K that maximizes `RDC`. If several values of K attain the same maximum `RDC`, the smallest such K is selected.

Monte Carlo p-values follow the rank-based convention used in **rflexscan**:

$$p = \frac{\mathrm{rank}}{M + 1},$$

where M is `simcount`. The overall p-value `P` compares the maximum `RDC` from the observed data with maxima from datasets simulated under the null Poisson model. Each null replication repeats the same candidate search and model-selection steps on a Poisson draw under the null expected counts. Cluster-level p-values `pval` compare each candidate cluster's scan statistic `stats` with statistics from the simulated datasets.

Value

A list of class `"multiflexscan"` with the following components:

call The matched call.

input A list with `coordinates` (centroid matrix), `case` (data frame of observed and expected counts), and `adj_mat` (binary adjacency matrix with zero diagonal).

cluster A list of candidate clusters returned by `rflexscan::runFleXScan()`, each extended with Monte Carlo `rank` and `pval`. Each element contains `area` (integer indices into the input regions), `name`, `max_dist`, `n_case`, `expected`, `RR`, `stats`, `pval`, and `rank`.

RDC Numeric vector of relative difference criterion values for $K = 0, 1, \ldots$ clusters.

neg2logLik, AIC, BIC, C Numeric vectors of model-selection criteria for $K = 0, 1, \ldots$ clusters.

nclust Selected number of clusters (index K with maximum `RDC`).

P Monte Carlo p-value for the overall clustering test.

rank Rank of the observed maximum `RDC` among simulated maxima.

null_dist Numeric vector of maximum `RDC` values from Monte Carlo replications.

setting A list recording analysis settings such as `simcount`, `maxclusters`, `clustersize`, `clusterradius`, `stattype`, `scanmethod`, and `clustertype`.

References

Tango T. and Takahashi K. (2005). A flexibly shaped spatial scan statistic for detecting clusters, International Journal of Health Geographics 4:11.

Kulldorff M. and Nagarwalla N. (1995). Spatial disease clusters: Detection and Inference. Statistics in Medicine 14:799-810.

Kulldorff M. (1997). A spatial scan statistic. Communications in Statistics: Theory and Methods, 26:1481-1496.

Tango T. (2008). A spatial scan statistic with a restricted likelihood ratio. Japanese Journal of Biometrics 29(2):75-95.

Takahashi K. and Shimadzu H. (2020). Detecting multiple spatial disease clusters: information criterion and scan statistic approach. International Journal of Health Geographics 19:33. doi:10.1186/s1294202000228y

Otani T. and Takahashi K. (2021). Flexible scan statistics for detecting spatial disease clusters: The rflexscan R package. Journal of Statistical Software 99:13. doi:10.18637/jss.v099.i13

See Also

rflexscan::rflexscan, summary.multiflexscan, plot.multiflexscan, choropleth

Examples

```
if (requireNamespace("spdep", quietly = TRUE)) {
  # load sample data (North Carolina SIDS data)
  library(spdep)
  data("nc.sids")

  # calculate the expected numbers of cases
  expected <- nc.sids$BIR74 * sum(nc.sids$SID74) / sum(nc.sids$BIR74)

  # small settings are used here to keep the example fast; increase
  # 'simcount', 'clustersize', and 'maxclusters' for practical analyses
  fit <- multiflexscan(
    x = nc.sids$x, y = nc.sids$y,
    observed = nc.sids$SID74,
    expected = expected,
    name = rownames(nc.sids),
    nb = ncCR85.nb,
    clustersize = 5,
    maxclusters = 3,
    simcount = 99,
    cores = 1
  )
  print(fit)
  summary(fit)
  plot(fit)
}
```

`nclusters` *Number of clusters selected by the information criterion*

Description

Number of clusters selected by the information criterion

Usage

```
nclusters(object)
```

Arguments

object	A multiflexscan object.

Value

Integer number of selected clusters.

See Also

pvalue(), clusters(), get_setting()

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
nclusters(fit)
```

---

nobs.multiflexscan *Number of regions in the analysis*

Description

Number of regions in the analysis

Usage

```
## S3 method for class 'multiflexscan'
nobs(object, ...)
```

Arguments

object	A multiflexscan object.
...	Additional arguments (ignored).

Value

Number of regions (rows) in the input data.

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
nobs(fit)
```

`plot.multiflexscan` *Plot method for the* `multiflexscan` *object. The neighborhood graph is drawn with* **igraph**, *following* `plot.rflexscan()`.

Description

Plot method for the `multiflexscan` object. The neighborhood graph is drawn with **igraph**, following `plot.rflexscan()`.

Usage

```
## S3 method for class 'multiflexscan'
plot(
  x,
  rank = seq_len(nclusters(x)),
  pval = 1,
  vertexsize = max(x$input$coordinates[, 1]) - min(x$input$coordinates[, 1]),
  xlab = colnames(x$input$coordinates)[1],
  ylab = colnames(x$input$coordinates)[2],
  xlim = c(min(x$input$coordinates[, 1]), max(x$input$coordinates[, 1])),
  ylim = c(min(x$input$coordinates[, 2]), max(x$input$coordinates[, 2])),
  col = palette(),
  frame_color = "gray40",
  vertex_color = "white",
  ...
)
```

Arguments

`x`	A `multiflexscan` object.
`rank`	Integer vector specifying the clusters to highlight.
`pval`	Maximum cluster-level p-value for a candidate cluster to be highlighted.
`vertexsize`	Size of region centroids in the igraph plot.

xlab	Label of the x-axis.
ylab	Label of the y-axis.
xlim	Limits of the x-axis.
ylim	Limits of the y-axis.
col	Colors used for highlighted clusters.
frame_color	Color used for neighborhood edges and vertex frames not highlighted as clusters.
vertex_color	Color used for region centroids that are not highlighted as clusters.
...	Additional arguments passed to igraph::plot.igraph().

Value

The input object, invisibly.

See Also

multiflexscan, choropleth, rflexscan::plot.rflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
plot(fit, xlab = "Grid x", ylab = "Grid y")
```

---

print.multiflexscan *Print multiflexscan object*

Description

Print method for multiflexscan objects.

Usage

```
## S3 method for class 'multiflexscan'
print(x, ...)
```

Arguments

x	A multiflexscan object to be printed.
...	Ignored.

Value

The input object, invisibly.

See Also

multiflexscan, summary.multiflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
print(fit)
```

---

print.summary.multiflexscan

Print summary of multiflexscan results

Description

Print summary of `multiflexscan` results to the terminal.

Usage

```
## S3 method for class 'summary.multiflexscan'
print(x, ...)
```

Arguments

`x`	A `summary.multiflexscan` object to be printed.
`...`	Ignored.

Value

The input object, invisibly.

See Also

multiflexscan, summary.multiflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
report <- summary(fit)
print(report)
```

---

pvalue *Overall Monte Carlo p-value*

Description

Overall Monte Carlo p-value

Usage

```
pvalue(object)
```

Arguments

object	A multiflexscan object.

Value

Numeric overall p-value for the multiple-cluster detection test.

See Also

nclusters(), clusters()

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
pvalue(fit)
```

summary.multiflexscan *Summarizing multiflexscan results*

Description

Summary method for multiflexscan objects.

Usage

```
## S3 method for class 'multiflexscan'
summary(object, ...)
```

Arguments

object	A multiflexscan object to be summarized.
...	Ignored.

Value

A list of class "summary.multiflexscan" with components call, total_areas, total_cases, cluster (data frame of per-cluster statistics), RDC, nclust, P, and setting.

See Also

multiflexscan, print.summary.multiflexscan

Examples

```
# A small synthetic grid; low simulation count keeps this example quick.
grid <- expand.grid(x = 1:5, y = 1:5)
nb <- lapply(seq_len(nrow(grid)), function(i) which(
  abs(grid$x - grid$x[i]) + abs(grid$y - grid$y[i]) == 1))
observed <- rep(10, 25)
observed[c(1, 2, 6, 7)] <- 30
fit <- multiflexscan(grid$x, grid$y, name = as.character(seq_len(25)),
  observed = observed, expected = rep(mean(observed), 25), nb = nb,
  scanmethod = "CIRCULAR", clustersize = 4, maxclusters = 3,
  simcount = 19, cores = 1, seed = 20261002)
summary(fit)
```

Index

AIC.multiflexscan, 3, *10*
as.data.frame.multiflexscan, 4, *7*

BIC.multiflexscan *(*AIC.multiflexscan*)*, 3

C_criterion *(*model_criteria*)*, 10
C_criterion(), *3*
choropleth, 5, *14*, *17*
clusters, 6
clusters(), *4*, *7*, *15*, *19*
coef.multiflexscan, *7*, 7

get_setting, 8
get_setting(), *15*

logLik(), *3*
logLik.multiflexscan, *4*, 9, *10*

model_criteria, 10
model_criteria(), *4*, *9*
multiflexscan, *6*, 11, *17, 18*, *20*
multiflexscan-package, 2

nclusters, 14
nclusters(), *7, 8*, *10*, *19*
nobs(), *9*
nobs.multiflexscan, 15

plot.multiflexscan, *6*, *14*, 16
print.multiflexscan, 17
print.summary.multiflexscan, 18, *20*
pvalue, 19
pvalue(), *8*, *15*

RDC *(*model_criteria*)*, 10
RDC(), *3*
rflexscan::plot.rflexscan, *17*
rflexscan::rflexscan, *14*

stats::AIC(), *3*, *9*
summary.multiflexscan, *14*, *18*, 20