Package ‘echelon’

August 21, 2025

Title The Echelon Analysis and the Detection of Spatial Clusters using Echelon Scan Method

Version 0.4.0

Description Functions for the echelon analysis proposed by Myers et al. (1997) <doi:10.1023/A:1018518327329>, and the detection of spatial clusters using echelon scan method proposed by Kurihara (2003) <doi:10.20551/jscswabun.15.2_171>.

License GPL-3

Depends R (>= 4.1.0)

Suggests sf, spData, spdep, mapview

Encoding UTF-8

RoxygenNote 7.3.2

LinkingTo Rcpp

Imports Rcpp

NeedsCompilation yes

Author Fumio Ishioka [aut, cre]

Maintainer Fumio Ishioka <fishioka@okayama-u.ac.jp>

Repository CRAN

Date/Publication 2025-08-21 05:00:02 UTC

Contents

echebin *Echelon spatial scan statistic based on Binomial model*

Description

The `echebin` function detects spatial clusters using the echelon spatial scan statistic with a Binomial model.

Usage

```
echebin(echelon.obj, cas, ctl, K = length(cas)/2, Kmin = 1, n.sim = 99,
        cluster.type = "high", cluster.legend.pos = "bottomleft",
        dendrogram = TRUE, cluster.info = FALSE, coo = NULL, ...)
```

Arguments

`echelon.obj` An object of class `echelon`. For details, see `echelon`.

`cas` A numeric (integer) vector of case counts. `NA` values are not allowed.

`ctl` A numeric (integer) vector of control counts. `NA` values are not allowed.

`K` Maximum cluster size. If `K` >= 1 (integer), the cluster size is limited to `K` regions. If 0 < `K` < 1, the cluster size is limited to `K` * 100% of the total population.

`Kmin` Minimum cluster size.

`n.sim` The number of Monte Carlo replications used for significance testing of detected clusters. If set to 0, significance is not assessed.

`cluster.type` A character string specifying the cluster type. If `"high"`, the detected clusters have high rates (hotspot). If `"low"`, the detected clusters have low rates (coldspot).

`cluster.legend.pos`
The location of the legend on the dendrogram. (See `legend` for details.)

`dendrogram` Logical. If TRUE, draws an echelon dendrogram with the detected clusters.

`cluster.info` Logical. If TRUE, returns detailed results of the detected clusters.

`coo` An array of (x, y) coordinates for the region centroids to plot a cluster map.

`...` Related to dendrogram drawing. (See the help for `echelon`)

Value

`clusters` Each detected cluster.

`scanned.regions`
A region list of all scanning processes.

`simulated.LLR` Monte Carlo samples of the log-likelihood ratio.

Note

The function `echebin` requires either `cas` or `ctl`.

Population is defined as the sum of `cas` and `ctl`.

Typical values of `n.sim` are 99, 999, 9999, ...

Author(s)

Fumio Ishioka

References

[1] Kulldorff M, Nagarwalla N. (1995). Spatial disease clusters: Detection and inference. *Statistics in Medicine*, **14**, 799–810.

[2] Kulldorff M. (1997). A spatial scan statistic. *Communications in Statistics: Theory and Methods*, **26**, 1481–1496.

See Also

`echelon` for the echelon analysis.

`echepoi` for cluster detection based on echelons using Poisson model.

`echenor` for cluster detection based on echelons using Normal model.

Examples

```
##Hotspot detection for non-white birth in North Carolina using echelon scan

#Load required packages and data
library(spData)
data("nc.sids")

#Non-white birth from 1974 to 1984 (case data)
nwb <- nc.sids$NWBIR74 + nc.sids$NWBIR79

#White birth from 1974 to 1984 (control data)
wb <- (nc.sids$BIR74 - nc.sids$NWBIR74) + (nc.sids$BIR79 - nc.sids$NWBIR79)

##Hotspot detection based on Binomial model
#Echelon analysis
SIDS.echelon <- echelon(x = nwb/wb, nb = ncCR85.nb, name = row.names(nc.sids))

#Basic cluster detection (significance not evaluated)
SIDS.clusters <- echebin(SIDS.echelon, cas = nwb, ctl = wb, K = 20,
  n.sim = 0, cluster.info = TRUE, main = "Hgih rate clusters", ens = FALSE)

#Significance assessment of clusters using Monte Carlo simulation
SIDS.clusters <- echebin(SIDS.echelon, cas = nwb, ctl = wb, K = 20,
  n.sim = 199, cluster.info = TRUE, main = "Hgih rate clusters", ens = FALSE)

text(SIDS.echelon$coord, labels = SIDS.echelon$regions.name,
  adj = -0.1, cex = 0.7)

#Detected clusters and neighbors map
#XY coordinates of each polygon centroid point
NC.coo <- cbind(nc.sids$lon, nc.sids$lat)
echebin(SIDS.echelon, cas = nwb, ctl = wb, K = 20,
  n.sim = 0, coo = NC.coo, dendrogram = FALSE)

#Load geospatial information for North Carolina
nc <- sf::st_read(system.file("shape/nc.shp", package = "sf"))

#Extract detected clusters
MLC <- SIDS.clusters$clusters[[1]]
Secondary <- SIDS.clusters$clusters[[2]]

#Assign colors to clusters for plotting
cluster.col <- rep(0, length(nwb))
cluster.col[MLC$regionsID] <- 2
cluster.col[Secondary$regionsID] <- 3

#Plot detected high-rate clusters on a simple map
plot(nc$geom, col = cluster.col,
  main = "Detected high rate clusters")
legend("bottomleft",
  legend = c(
    paste("1- p-value:", MLC$p),
    paste("2- p-value:", Secondary$p)
  ),
  text.col = c(2, 3)
)

#Interactive map visualization with mapview
library(mapview)
nc$cluster.col <- cluster.col
mapview(nc, zcol = "cluster.col",
  col.regions=c("white", "red", "green"),
  label = "NAME", legend=FALSE)
```

echelon *Echelon analysis for spatial data*

Description

The echelon function divides the study area into structural entities, called *’echelons’*, based on neighbor information and draws a dendrogram.

Usage

```
echelon(x, nb, dendrogram = TRUE, name = NULL,
      main = NULL, ylab = NULL, yaxes = TRUE, ylim = NULL,
      xaxes = FALSE, xdper = c(0, 1), dmai = NULL,
      col = 1, lwd = 1, symbols = 4, cex.symbols = 1, col.symbols = 4,
      ens = TRUE, adj.ens = 1, cex.ens = 0.8, col.ens = 1,
      profiles = FALSE, nb.check = TRUE)
```

Arguments

`x` A numeric vector containing data values.

`nb` Neighbor information data: an object of class `nb` or a weights matrix.

`name` Region names. if NULL, it is assigned `seq_along(x)`.

`dendrogram` Logical. if TRUE, draws an echelon dendrogram.

`main` Related to dendrogram drawing. The main title for the dendrogram.

`ylab` Related to dendrogram drawing. The title for the y-axis.

`yaxes` Related to dendrogram drawing. Logical. if TRUE, draws the y-axis.

`ylim` Related to dendrogram drawing. If not specified, the y-axis scale is set to `c(min, max)`.

`xaxes` Related to dendrogram drawing. Logical. if TRUE, draws the x-axis.

`xdper` Related to dendrogram drawing. The percentage of the x-axis to display, specified in *[0, 1]*.

`dmai` Related to dendrogram drawing. A numeric vector of the form `c(bottom, left, top, right)` specifying margin sizes in inches. Default is `c(0.4, 0.8, 0.3, 0.01)`.

`col` Related to dendrogram drawing. The line color of the dendrogram.

`lwd` Related to dendrogram drawing. The line width of the dendrogram.

`symbols` Related to dendrogram drawing. An integer specifying a symbol or a single character. If integer, it corresponds to `pch` in `par`.

`cex.symbols` Related to dendrogram drawing. A magnification factor for the plotting symbols.

`col.symbols` Related to dendrogram drawing. The color for the plotting symbols.

`ens` Related to dendrogram drawing. Logical. if TRUE, draw the labels of echelon numbers.

`adj.ens` Related to dendrogram drawing. Adjusts the position of echelon number labels (see `text` for 'adj').

`cex.ens` Related to dendrogram drawing. A magnification factor for the echelon number labels.

`col.ens` Related to dendrogram drawing. The color for the echelon number labels.

`profiles` Logical. If TRUE, returns the echelon profiles result (see [2] for details).

`nb.check` Logical. if TRUE, checks for errors in the neighbor information data.

Value

The `echelon` function returns an object of class `echelon`, which contains the following components:

`Table` A summary of each echelon.

`Echelons` The regions that make up each echelon.

Note

Any NA values in x are replaced with the minimum value of x.

The functions Sf::st_read and spdep::poly2nb are helpful for creating the object specified in the nb argument.

Author(s)

Fumio Ishioka

References

[1] Myers, W.L., Patil, G.P. and Joly, K. (1997). Echelon approach to areas of concern in synoptic regional monitoring. *Environmental and Ecological Statistics*, **4**, 131–152.

[2] Kurihara, K., Myers, W.L. and Patil, G.P. (2000) Echelon analysis of the relationship between population and land cover patter based on remote sensing data. *Community ecology*, **1**, 103–122.

See Also

echepoi, echebin and echenor for cluster detection based on echelons.

Examples

```
##Echelon analysis for one-dimensional data with 25 regions
#A weights matrix
one.nb <- matrix(0,25,25)
one.nb[1,2] <- 1
for(i in 2:24) one.nb[i,c(i-1,i+1)] <- c(1,1)
one.nb[25,24] <- 1

#25 random values
one.dat <- runif(25) * 10

#Echelon analysis
echelon(x = one.dat, nb = one.nb)


##Echelon analysis for SIDS data for North Carolina
#Mortality rate per 1,000 live births from 1974 to 1984
library(spData)
data("nc.sids")
SIDS.cas <- nc.sids$SID74 + nc.sids$SID79
SIDS.pop <- nc.sids$BIR74 + nc.sids$BIR79
SIDS.rate <- SIDS.cas * 1000 / SIDS.pop

#Echelon analysis
SIDS.echelon <- echelon(x = SIDS.rate, nb = ncCR85.nb, name = row.names(nc.sids),
  symbols = 12, cex.symbols = 1.5, ens = FALSE)
text(SIDS.echelon$coord, labels = SIDS.echelon$regions.name,
  adj = -0.1, cex = 0.7)

#Echelon Profiles
```

```
echelon(x = SIDS.rate, nb = ncCR85.nb, profiles = TRUE)
```

echenor *Echelon spatial scan statistic based on Normal model*

Description

The `echenor` function detects spatial clusters using the echelon spatial scan statistic with a Normal model.

Usage

```
echenor(echelon.obj, val, weight = NULL, K = length(val)/2, Kmin = 2, n.sim = 99,
        cluster.type = "high", cluster.legend.pos = "bottomleft",
        dendrogram = TRUE, cluster.info = FALSE, coo = NULL, ...)
```

Arguments

`echelon.obj` An object of class `echelon`. For details, see `echelon`.

`val` A numeric vector of observed values, which may be positive or negative. `NA` values are not allowed.

`weight` A numeric vector of weighted values (must be positive). If `NULL` (the default), all weights are set to 1. `NA` values are not allowed.

`K` Maximum cluster size. If `K` >= 2 (integer), the cluster size is limited to `K` regions.

`Kmin` Minimum cluster size. Must be at least 2, due to the use of a permutation-based Monte Carlo test.

`n.sim` The number of Monte Carlo replications used for significance testing of detected clusters. If set to 0, significance is not assessed.

`cluster.type` A character string specifying the cluster type. If `"high"`, the detected clusters have high rates (hotspot). If `"low"`, the detected clusters have low rates (coldspot).

`cluster.legend.pos`
The location of the legend on the dendrogram. (See `legend` for details.)

`dendrogram` Logical. If TRUE, draws an echelon dendrogram with the detected clusters.

`cluster.info` Logical. If TRUE, returns detailed results of the detected clusters.

`coo` An array of (x, y) coordinates for the region centroids to plot a cluster map.

`...` Related to dendrogram drawing. (See the help for `echelon`)

Value

`clusters` Each detected cluster.

`scanned.regions`
A region list of all scanning processes.

`simulated.LLR` Monte Carlo samples of the log-likelihood ratio.

Note

Typical values of n.sim are 99, 999, 9999, ...

Author(s)

Fumio Ishioka

References

[1] Kulldorff M, Huang L, and Konty K. (2009). A scan statistic for continuous data based on the normal probability model. *International Journal of Health Geographics*, **8**, 58.

[2] Huang L, Tiwari R, Zuo J, Kulldorff M, and Feuer E. (2009) Weighted normal spatial scan statistic for heterogeneous population data. *Journal of the American Statistical Association*, **104**, 886–898.

See Also

echelon for the echelon analysis.

echepoi for cluster detection based on echelons using Poisson model.

echebin for cluster detection based on echelons using Binomial model.

Examples

```
##Hotspot detection for predicting SIDS rate in North Carolina using echelon scan

#Load required packages and data
library(spData)
data("nc.sids")
SIDS.cas <- nc.sids$SID74 + nc.sids$SID79
SIDS.pop <- nc.sids$BIR74 + nc.sids$BIR79
SIDS.nwpop <- nc.sids$NWBIR74 + nc.sids$NWBIR79
SIDS.rate <- SIDS.cas * 1000 / SIDS.pop

#Fit a linear model: SIDS rate explained by proportion of non-white births
res <- lm(SIDS.rate ~ I(SIDS.nwpop / SIDS.pop))
summary(res)

#Predicted values and reliability weights (inverse of standard error)
pred <- predict(res, newdata = nc.sids, se.fit = TRUE)
V <- res$fitted.values
W <- 1 / (pred$se.fit + 1e-6)

##Hotspot detection based on Normal model
#Echelon analysis
SIDS.echelon <- echelon(x = V, nb = ncCR85.nb, name = row.names(nc.sids))

#Basic cluster detection (significance not evaluated)
SIDS.clusters <- echenor(SIDS.echelon, val = V, weight = W, K = 20,
  n.sim = 0, cluster.info = TRUE, main = "Hgih value clusters", ens = FALSE)
```

```
#Significance assessment of clusters using Monte Carlo simulation
SIDS.clusters <- echenor(SIDS.echelon, val = V, weight = W, K = 20,
  n.sim = 199, cluster.info = TRUE, main = "Hgih value clusters", ens = FALSE)

text(SIDS.echelon$coord, labels = SIDS.echelon$regions.name,
  adj = -0.1, cex = 0.7)

#Load geospatial information for North Carolina
nc <- sf::st_read(system.file("shape/nc.shp", package = "sf"))

#Extract detected clusters
MLC <- SIDS.clusters$clusters[[1]]
Secondary <- SIDS.clusters$clusters[[2]]

#Assign colors to clusters for plotting
cluster.col <- rep(0, length(V))
cluster.col[MLC$regionsID] <- 2
cluster.col[Secondary$regionsID] <- 3

#Plot detected high-value clusters on a simple map
plot(nc$geom, col = cluster.col,
  main = "Detected high value clusters")
legend("bottomleft",
  legend = c(
    paste("1- p-value:", MLC$p),
    paste("2- p-value:", Secondary$p)
  ),
  text.col = c(2, 3)
)

#Interactive map visualization with mapview
library(mapview)
nc$cluster.col <- cluster.col
mapview(nc, zcol = "cluster.col",
  col.regions=c("white", "red", "green"),
  label = "NAME", legend=FALSE)
```

echepoi *Echelon spatial scan statistic based on Poisson model*

Description

The echepoi function detects spatial clusters using the echelon spatial scan statistic with a Poisson model.

Usage

```
echepoi(echelon.obj, cas, pop = NULL, ex = NULL, K = length(cas)/2, Kmin = 1, n.sim = 99,
        cluster.type = "high", cluster.legend.pos = "bottomleft",
        dendrogram = TRUE, cluster.info = FALSE, coo = NULL, ...)
```

Arguments

`echelon.obj` An object of class `echelon`. For details, see `echelon`.

`cas` A numeric (integer) vector of case counts. `NA` values are not allowed.

`pop` A numeric (integer) vector for population. `NA` values are not allowed.

`ex` A numeric vector for expected case counts. `NA` values are not allowed.

`K` Maximum cluster size. If `K` >= 1 (integer), the cluster size is limited to `K` regions. If 0 < `K` < 1, the cluster size is limited to `K` * 100% of the total population.

`Kmin` Minimum cluster size.

`n.sim` The number of Monte Carlo replications used for significance testing of detected clusters. If set to 0, significance is not assessed.

`cluster.type` A character string specifying the cluster type. If `"high"`, the detected clusters have high rates (hotspot). If `"low"`, the detected clusters have low rates (coldspot).

`cluster.legend.pos` The location of the legend on the dendrogram. (See `legend` for details.)

`dendrogram` Logical. If TRUE, draws an echelon dendrogram with the detected clusters.

`cluster.info` Logical. If TRUE, returns detailed results of the detected clusters.

`coo` An array of (x, y) coordinates for the region centroids to plot a cluster map.

`...` Related to dendrogram drawing. (See the help for `echelon`)

Value

`clusters` Each detected cluster.

`scanned.regions` A region list of all scanning processes.

`simulated.LLR` Monte Carlo samples of the log-likelihood ratio.

Note

The function `echepoi` requires either `pop` or `ex`.

Typical values of `n.sim` are 99, 999, 9999, ...

Author(s)

Fumio Ishioka

References

[1] Kulldorff M. (1997). A spatial scan statistic. *Communications in Statistics: Theory and Methods*, **26**, 1481–1496.

[2] Ishioka F, Kawahara J, Mizuta M, Minato S, and Kurihara K. (2019) Evaluation of hotspot cluster detection using spatial scan statistic based on exact counting. *Japanese Journal of Statistics and Data Science*, **2**, 241–262.

See Also

echelon for the echelon analysis.

echebin for cluster detection based on echelons using Binomial model.

echenor for cluster detection based on echelons using Normal model.

Examples

```
##Hotspot detection for SIDS cases in North Carolina using echelon scan

#Load required packages and data
library(spData)
data("nc.sids")
SIDS.cas <- nc.sids$SID74 + nc.sids$SID79
SIDS.pop <- nc.sids$BIR74 + nc.sids$BIR79
SIDS.rate <- SIDS.cas * 1000 / SIDS.pop

##Hotspot detection based on Poisson model
#Echelon analysis
SIDS.echelon <- echelon(x = SIDS.rate, nb = ncCR85.nb, name = row.names(nc.sids))

#Basic cluster detection (significance not evaluated)
SIDS.clusters <- echepoi(SIDS.echelon, cas = SIDS.cas, pop = SIDS.pop, K = 20,
  n.sim = 0, cluster.info = TRUE, main = "Hgih rate clusters", ens = FALSE)

#Significance assessment of clusters using Monte Carlo simulation
SIDS.clusters <- echepoi(SIDS.echelon, cas = SIDS.cas, pop = SIDS.pop, K = 20,
  n.sim = 199, cluster.info = TRUE, main = "Hgih rate clusters", ens = FALSE)

text(SIDS.echelon$coord, labels = SIDS.echelon$regions.name,
  adj = -0.1, cex = 0.7)

#Detected clusters and neighbors map
#XY coordinates of each polygon centroid point
NC.coo <- cbind(nc.sids$lon, nc.sids$lat)
echepoi(SIDS.echelon, cas = SIDS.cas, pop = SIDS.pop, K = 20,
  n.sim = 0, coo = NC.coo, dendrogram = FALSE)

#Load geospatial information for North Carolina
nc <- sf::st_read(system.file("shape/nc.shp", package = "sf"))

#Extract detected clusters
MLC <- SIDS.clusters$clusters[[1]]
Secondary <- SIDS.clusters$clusters[[2]]

#Assign colors to clusters for plotting
cluster.col <- rep(0, length(SIDS.rate))
cluster.col[MLC$regionsID] <- 2
cluster.col[Secondary$regionsID] <- 3

#Plot detected high-rate clusters on a simple map
plot(nc$geom, col = cluster.col,
  main = "Detected high rate clusters")
legend("bottomleft",
  legend = c(
    paste("1- p-value:", MLC$p),
    paste("2- p-value:", Secondary$p)
  ),
  text.col = c(2, 3)
)

#Interactive map visualization with mapview
library(mapview)
nc$cluster.col <- cluster.col
mapview(nc, zcol = "cluster.col",
  col.regions=c("white", "red", "green"),
  label = "NAME", legend=FALSE)
```

Index