Package 'scanstatistics'

September 29, 2016

Type Package

Title Space-Time Anomaly Detection using Scan Statistics

Description Detection of anomalous space-time clusters using the scan statistics methodology. Focuses on prospective surveillance of data streams, scanning for clusters with ongoing anomalies. Hypothesis testing is made possible by the generation of Monte Carlo p-values.

Version 0.1.0

Date 2016-09-28

Maintainer Benjamin Kjellson <benjak@math.su.se>

Encoding UTF-8

License GPL (>= 3)

URL https://github.com/BenjaK/scanstatistics

BugReports https://github.com/BenjaK/scanstatistics/issues

Depends data.table, R (>= 2.10)

Imports gamlss.dist, magrittr, plyr, sets

Suggests doParallel, foreach, ggplot2, knitr, MASS, pscl, purrr, reshape2, rmarkdown, sp, testthat

VignetteBuilder knitr

RoxygenNote 5.0.1

ByteCompile true

NeedsCompilation no

Author Benjamin Kjellson [aut, cre]

Repository CRAN

Date/Publication 2016-09-29 11:37:32

R topics documented:

`coords_to_knn` *Get the k nearest neighbors for each location, given its coordinates.*

Description

Get the k nearest neighbors for each location, including the location itself. This function calls `dist`, so the options for the distance measure used is the same as for that one. Distances are calculated between rows.

Usage

```
coords_to_knn(x, k = min(10, nrow(x)), method = "euclidean", p = 2)
```

Arguments

`x`	a numeric matrix, data frame or `"dist"` object.
`k`	The number of nearest neighbors, counting the location itself.
`method`	the distance measure to be used. This must be one of `"euclidean"`, `"maximum"`, `"manhattan"`, `"canberra"`, `"binary"` or `"minkowski"`. Any unambiguous substring can be given.
`p`	The power of the Minkowski distance.

Value

An integer matrix of the k nearest neighbors for each location. Each row corresponds to a location, with the first element of each row being the location itself. Locations are encoded as integers.

Examples

```
x <- matrix(c(0, 0,
              1, 0,
              2, 1,
              0, 4,
              1, 3),
            ncol = 2, byrow = TRUE)
plot(x)
coords_to_knn(x)
```

`dist_to_knn` *Given a distance matrix, find the k nearest neighbors.*

Description

Given a distance matrix, calculate the k nearest neighbors of each location, including the location itself. The matrix should contain only zeros on the diagonal, and all other elements should be positive.

Usage

```
dist_to_knn(x, k = min(10, nrow(x)))
```

Arguments

`x`	A (square) distance matrix. Elements should be non-negative and the diagonal zeros, but this is not checked.
`k`	The number of nearest neighbors, counting the location itself.

Value

A matrix of integers, row i containing the k nearest neighbors of location i, including itself.

Examples

```
x <- matrix(c(0, 0,
              1, 0,
              2, 1,
              0, 4,
              1, 3),
            ncol = 2, byrow = TRUE)
d <- dist(x, diag = TRUE, upper = TRUE)
dist_to_knn(d, k = 3)
```

`flexible_zones` *Computes the flexibly shaped zones as in Tango (2005).*

Description

Given a matrix of k nearest neighbors and an adjacency matrix for the locations involved, produces the set of flexibly shaped zones as a list of integer vectors. The locations in these zones are all connected, in the sense that any location in the zone can be reached from another by traveling through adjacent locations within the zone.

Usage

```
flexible_zones(k_nearest, adjacency_matrix, .parallel = FALSE,
  .paropts = NULL)
```

Arguments

`k_nearest`	An integer matrix of the k nearest neighbors for each location. Each row corresponds to a location, with the first element of each row being the location itself. Locations should be encoded as integers.
`adjacency_matrix`	A boolean matrix, with element (i, j) set to TRUE if location j is adjacent to location i.
`.parallel`	if `TRUE`, apply function in parallel, using parallel backend provided by foreach
`.paropts`	a list of additional options passed into the `foreach` function when parallel computation is enabled. This is important if (for example) your code relies on external data or packages: use the `.export` and `.packages` arguments to supply them so that all cluster nodes have the correct environment set up for computing.

Value

A list of integer vectors.

References

Tango, T. & Takahashi, K. (2005), *A flexibly shaped spatial scan statistic for detecting clusters*, International Journal of Health Geographics 4(1).

Examples

```
A <- matrix(c(0,1,0,0,0,0,
              1,0,1,0,0,0,
              0,1,0,0,0,0,
              0,0,0,0,1,0,
              0,0,0,1,0,0,
              0,0,0,0,0,0),
              nrow = 6, byrow = TRUE) == 1
```

```
nn <- matrix(as.integer(c(1,2,3,4,5,6,
                          2,1,3,4,5,6,
                          3,2,1,4,5,6,
                          4,5,1,6,3,2,
                          5,4,6,1,3,2,
                          6,5,4,1,3,2)),
                          nrow = 6, byrow = TRUE)
flexible_zones(nn, A)
```

`get_zone` *Extract a zone from the set of all zones.*

Description

Extract zone number n from the set of all zones.

Usage

```
get_zone(n, zones)
```

Arguments

`n`	An integer; the number of the zone you wish to retrieve.
`zones`	A list of integer vectors, representing the set of all zones.

Value

An integer vector.

Examples

```
zones <- list(1L, 2L, 3L, 1:2, c(1L, 3L), c(2L, 3L))
get_zone(4, zones)
```

`knn_zones` *Find the increasing subsets of k nearest neighbors for all locations.*

Description

Returns the set of increasing nearest neighbor sets for all locations, as a list of integer vectors. That is, for each location the list returned contains one vector containing the location itself, another containing the location and its nearest neighbor, and so on, up to the vector containing the location and its $k - 1$ nearest neighbors.

Usage

```
knn_zones(k_nearest, .parallel = FALSE, .paropts = NULL)
```

Arguments

k_nearest	An integer matrix of with k columns and as many rows as locations. The first element of each row is the integer encoding the location (and equal to the row number); the following elements are the $k - 1$ nearest neighbors in ascending order of distance.
.parallel	if TRUE, apply function in parallel, using parallel backend provided by foreach
.paropts	a list of additional options passed into the foreach function when parallel computation is enabled. This is important if (for example) your code relies on external data or packages: use the .export and .packages arguments to supply them so that all cluster nodes have the correct environment set up for computing.

Value

A list of integer vectors.

Examples

```
nn <- matrix(c(1L, 2L, 4L, 3L, 5L,
               2L, 1L, 3L, 4L, 5L,
               3L, 2L, 4L, 1L, 5L,
               4L, 1L, 2L, 3L, 5L,
               5L, 3L, 4L, 2L, 1L),
               ncol = 5, byrow = TRUE)
knn_zones(nn[, 1:3])
```

---

mc_pvalue *Calculate the Monte Carlo p-value for a scan statistic.*

Description

Given an observed scan statistic λ^* and a vector of replicate scan statistics λ_i, $i = 1, \ldots, R$, calculate the Monte carlo p-value as

$$\frac{1 + \sum_{i=1}^{R} \mathrm{I}(\lambda_i > \lambda^*)}{1 + R}$$

The function is vectorized, so multiple p-values can be calculated if several scan statistics (e.g. statistics from secondary clusters) are supplied.

Usage

```
mc_pvalue(observed, replicates)
```

Arguments

observed	A scalar containing the observed value of the scan statistic, or a vector of observed values from secondary clusters.
replicates	A vector of Monte Carlo replicates of the scan statistic.

Value

The p-value or p-values corresponding to the observed scan statistic(s).

`NM_geo` *Longitude and latitude of New Mexico county seats.*

Description

A dataset containing the longitude and latitude of the county seats of New Mexico, except for Cibola county.

Usage

```
NM_geo
```

Format

A data table with 32 rows and 4 variables:

county The name of the county. Of class character.

seat The name of the county seat, i.e. the administrative center or seat of government. Of class factor with 32 levels.

long The longitude of the county seat. Of class numeric.

lat The latitude of the county seat. Of class numeric.

Source

https://en.wikipedia.org/wiki/List_of_counties_in_New_Mexico

`NM_map` *Data to plot the counties of New Mexico.*

Description

Map data for New Mexico. Was created using `ggplot2::map_data`.

Usage

```
NM_map
```

Format

A data table with 867 rows and 7 variables:

long Longitude of county polygon corner. Of class numeric.

lat Latitude of county polygon corner. Of class numeric.

group Grouping by county. Of class numeric.

order Order of the polygon corners. Of class integer.

region Region is "new mexico" for all rows. Of class character.

subregion The county name (with spaces). Of class character.

county The county name (no spaces). Of class factor with 33 levels, the county Cibola places last.

`NM_popcas` *Population and brain cancer cases in New Mexico counties during 1973–1991.*

Description

A dataset containing the population count and number of brain cancer cases in the counties of New Mexico during the years 1973–1991. The population numbers are interpolations from the censuses conducted in 1973, 1982, and 1991. Interpolations were done using a quadratic function of time. Thus the year-to-year changes are overly smooth but match the census numbers in the three years mentioned.

Usage

```
NM_popcas
```

Format

A data table with 608 rows and 4 variables:

year The year the cases were recorded. Of class integer.

county The name of the county. Of class factor with 32 levels.

population The population in that county and year. Of class numeric.

count The number of brain cancer cases in that county and year. Of class integer.

scanstatistics *scanstatistics: Space-time anomaly detection using scan statistics.*

Description

The scanstatistics package provides two categories of important functions: data preparation functions, and the scan statistics themselves.

Data preparation functions

These functions prepare your data for use. In particular, it helps you define the *zones* which will be considered by the scan statistics.

Scan statistics

These are the functions used for space-time anomaly detection. Scan statistic functions for univariate space-time data have a name that begins with `scan_` and functions for multivariate space-time data have a name that begins with `mscan_`.

scan_negbin *Calculate the negative binomial scan statistic.*

Description

Calculate the expectation-based negative binomial scan statistic by supplying a `data.table` of observed counts and pre-computed distribution parameters for each location and time. A p-value for the observed scan statistic can be obtained by Monte Carlo simulation.

Usage

```
scan_negbin(table, zones, n_mcsim = 0, version = "ordinary")
```

Arguments

`table`	A `data.table` with columns `location, duration, mu,theta, count`. The `location` column should consist of integers that are unique to each location. The `duration` column should also consist of integers, starting at 1 for the most recent time period and increasing in reverse chronological order. A negative binomial distribution parametrized by μ and θ (columns `mu` and `theta` respectively) has expected value μ and variance $\mu+\mu^2/\theta$. The parameter θ is referred to as the `size` in `NegBinomial`, and `theta` in `negative.binomial`.
`zones`	A `set` of zones, each zone itself a set containing one or more locations of those found in `table`.
`n_mcsim`	A non-negative integer; the number of replicate scan statistics to generate in order to calculate a p-value.
`version`	Which version of the negative binomial score scan statistic to calculate: either "ordinary" (default) or "increasing". See details.

Details

For the expectation-based negative binomial scan statistic (Tango et al., 2011), the null hypothesis of no anomaly holds that the count observed at each location i and duration t (the number of time periods before present) has a negative binomial distribution with expected value μ_{it} and dispersion parameter θ_{it}:

$$H_0 : Y_{it} \sim \text{NegBin}(\mu_{it}, \theta_{it}).$$

This holds for all locations $i = 1, \ldots, m$ and all durations $t = 1, \ldots, T$, with T being the maximum duration considered. The alternative hypothesis depends on the version used: if `version == "ordinary"`, then the alternative hypothesis states that there is a space-time window W consisting of a spatial zone $Z \subset \{1, \ldots, m\}$ and a time window $D \subseteq \{1, \ldots, T\}$ such that the counts in this window have their expected values inflated by a factor $q_W > 1$ compared to the null hypothesis:

$$H_1 : Y_{it} \sim \text{NegBin}(q_W \mu_{it}, \theta_{it}), \quad (i, t) \in W.$$

If `version == "increasing"`, q_W is instead increasing over time (decreasing with `duration`). For locations and durations outside of this window, counts are assumed to be distributed as under the null hypothesis. The sets Z considered are those specified in the argument `zones`, while the maximum duration T is taken as the maximum value in the column `duration` of the input `table`. For each space-time window W considered, a score statistic is computed using the score function and Fisher information under the null hypothesis of no anomaly. The scan statistic is calculated as the maximum of these quantities over all space-time windows. Point estimates of the parameters μ_{it} and θ_{it} must be specified in the column `mu` and `theta` of the argument `table` before this function is called.

Value

An object of class `scanstatistics`. It has the following fields:

observed A `data.table` containing the value of the statistic calculated for each zone-duration combination, for the observed data. The scan statistic is the maximum value of these calculated statistics.

replicated A numeric vector of length `n_mcsim` containing the values of the scanstatistics calculated by Monte Carlo simulation.

mlc A `data.table` containing the zone, duration, and scanstatistic.

pvalue The p-value calculated from Monte Carlo replications.

distribution The assumed distribution of the data; "negative binomial" in this case.

type The type of scan statistic; "Expectation-based" in this case.

zones The set of zones that was passed to the function as input.

n_locations The number of locations in the data.

n_zones The number of zones.

max_duration The maximum outbreak/event/anomaly duration considered.

References

Tango, T., Takahashi, K. & Kohriyama, K. (2011), *A space-time scan statistic for detecting emerging outbreaks*, Biometrics 67(1), 106–115.

Examples

```
# Simple example
set.seed(1)
table <- scanstatistics:::create_table(list(location = 1:4, duration = 1:4),
                                       keys = c("location", "duration"))
table[, mu := 3 * location]
table[, theta := 2]
table[, count := rnbinom(.N, mu = mu, size = theta)]
table[location %in% c(1, 4) & duration < 3,
      count :=  rnbinom(.N, mu = 2 * mu, size = theta)]
zones <- scanstatistics:::powerset_zones(4)
result1 <- scan_negbin(table, zones, 100, "ordinary")
result2 <- scan_negbin(table, zones, 100, "increasing")
```

`scan_poisson` *Calculate the Poisson scan statistic.*

Description

Calculate the expectation-based Poisson scan statistic by supplying a `data.table` of observed counts and pre-computed expected value parameters for each location and time. A p-value for the observed scan statistic can be obtained by Monte Carlo simulation.

Usage

```
scan_poisson(table, zones, n_mcsim = 0)
```

Arguments

`table`	A `data.table` with columns `location, duration, count, mu`. The `location` column should consist of integers that are unique to each location. The `duration` column should also consist of integers, starting at 1 for the most recent time period and increasing in reverse chronological order. The column `mu` should contain the estimated Poisson expected value parameter.
`zones`	A `set` of zones, each zone itself a set containing one or more locations of those found in `table`.
`n_mcsim`	A non-negative integer; the number of replicate scan statistics to generate in order to calculate a p-value.

Details

For the expectation-based Poisson scan statistic, the null hypothesis of no anomaly holds that the count observed at each location i and duration t (the number of time periods before present) is Poisson-distributed with expected value μ_{it}:

$$H_0 : Y_{it} \sim \text{Poisson}(\mu_{it}),$$

for all locations $i = 1, \ldots, m$ and all durations $t = 1, \ldots, T$, with T being the maximum duration considered. Under the alternative hypothesis, there is a space-time window W consisting of a spatial zone $Z \subset \{1, \ldots, m\}$ and a time window $D \subseteq \{1, \ldots, T\}$ such that the counts in that window have their expected values inflated by a factor $q_W > 1$ compared to the null hypothesis:

$$H_1 : Y_{it} \sim \text{Poisson}(q_W \mu_{it}), \quad (i,t) \in W.$$

For locations and durations outside of this window, counts are assumed to be distributed as under the null hypothesis. The sets Z considered are those specified in the argument `zones`, while the maximum duration T is taken as the maximum value in the column `duration` of the input `table`. For each space-time window W considered, (the log of) a likelihood ratio is computed using the distributions under the alternative and null hypotheses, and the expectation-based Poisson scan statistic is calculated as the maximum of these quantities over all space-time windows. Point estimates of the parameters μ_{it} must be specified in the column `mu` of the argument `table` before this function is called.

Value

An object of class `scanstatistics`. It has the following fields:

observed A `data.table` containing the value of the statistic calculated for each zone-duration combination, for the observed data. The scan statistic is the maximum value of these calculated statistics.

replicated A numeric vector of length `n_mcsim` containing the values of the scanstatistics calculated by Monte Carlo simulation.

mlc A `data.table` containing the zone, duration, and scanstatistic.

pvalue The p-value calculated from Monte Carlo replications.

distribution The assumed distribution of the data; "Poisson" in this case.

type The type of scan statistic; "Expectation-based" in this case.

zones The set of zones that was passed to the function as input.

n_locations The number of locations in the data.

n_zones The number of zones.

max_duration The maximum anomaly duration considered.

Examples

```
# Simple example
set.seed(1)
table <- scanstatistics:::create_table(list(location = 1:4, duration = 1:4),
                                       keys = c("location", "duration"))
table[, mu := 3 * location]
table[, count := rpois(.N, mu)]
table[location %in% c(1, 4) & duration < 3, count := rpois(.N, 2 * mu)]
zones <- scanstatistics:::powerset_zones(4)
result <- scan_poisson(table, zones, 100)
result
```

scan_zip *Calculate the ZIP scan statistic.*

Description

Calculate the expectation-based zero-inflated Poisson scan statistic by supplying a `data.table` of observed counts and pre-computed expected values and structural zero probabilities for each location and time. A p-value for the observed scan statistic can be obtained by Monte Carlo simulation.

Usage

```
scan_zip(table, zones, n_mcsim = 0, ...)
```

Arguments

`table`	A `data.table` with columns `location`, `duration`, `count`, `mu`, `p`. The `location` column should consist of integers that are unique to each location. The `duration` column should also consist of integers, starting at 1 for the most recent time period and increasing in reverse chronological order. The column `mu` should contain the estimated Poisson expected value parameters, and the column `p` the estimated structural zero probabilities.
`zones`	A `set` of zones, each zone itself a set containing one or more locations of those found in `table`.
`n_mcsim`	A non-negative integer; the number of replicate scan statistics to generate in order to calculate a p-value.
`...`	Arguments passed to internal functions. Arguments that can be passed here are `d`, the initial value for the excess zero indicators (default is 0.5), and `tol`, the threshold for the absolute convergence criterion (default is 0.01).

Details

For the expectation-based zero-inflated Poisson scan statistic (Kjellson 2015), the null hypothesis of no anomaly holds that the count observed at each location i and duration t (the number of time periods before present) has a zero-inflated Poisson distribution with expected value parameter μ_{it} and structural zero probability p_{it}:

$$H_0 : Y_{it} \sim \text{ZIP}(\mu_{it}, p_{it}).$$

This holds for all locations $i = 1, \ldots, m$ and all durations $t = 1, \ldots, T$, with T being the maximum duration considered. Under the alternative hypothesis, there is a space-time window W consisting of a spatial zone $Z \subset \{1, \ldots, m\}$ and a time window $D \subseteq \{1, \ldots, T\}$ such that the counts in that window have their Poisson expected value parameters inflated by a factor $q_W > 1$ compared to the null hypothesis:

$$H_1 : Y_{it} \sim \text{ZIP}(q_W \mu_{it}, p_{it}), \quad (i, t) \in W.$$

For locations and durations outside of this window, counts are assumed to be distributed as under the null hypothesis. The sets Z considered are those specified in the argument `zones`, while the maximum duration T is taken as the maximum value in the column `duration` of the input `table`.

For each space-time window W considered, (the log of) a likelihood ratio is computed using the distributions under the alternative and null hypotheses, and the expectation-based Poisson scan statistic is calculated as the maximum of these quantities over all space-time windows. The expectation-maximization (EM) algorithm is used to obtain maximum likelihood estimates. Point estimates of the parameters μ_{it} must be specified in the column `mu` of the argument `table` before this function is called.

Value

An object of class `scanstatistics`. It has the following fields:

observed A `data.table` containing the value of the statistic calculated for each zone-duration combination, for the observed data. The scan statistic is the maximum value of these calculated statistics.

replicated A numeric vector of length `n_mcsim` containing the values of the scanstatistics calculated by Monte Carlo simulation.

mlc A `data.table` containing the zone, duration, and scanstatistic.

pvalue The p-value calculated from Monte Carlo replications.

distribution The assumed distribution of the data; "zero-inflated Poisson" in this case.

type The type of scan statistic; "Expectation-based" in this case.

zones The set of zones that was passed to the function as input.

n_locations The number of locations in the data.

n_zones The number of zones.

max_duration The maximum anomaly duration considered.

References

Kjellson, B. (2015), *Spatiotemporal Outbreak Detection: A Scan Statistic Based on the Zero-Inflated Poisson Distribution*, (Master Thesis, Stockholm University), Link to PDF.

Examples

```
# Simple example
set.seed(1)
table <- scanstatistics:::create_table(list(location = 1:4, duration = 1:4),
                                       keys = c("location", "duration"))
table[, mu := 3 * location]
table[, p := runif(.N, 0, 0.3)]
table[, count := gamlss.dist::rZIP(.N, mu = mu, sigma = p)]
table[location %in% c(1, 4) & duration < 3,
      count := gamlss.dist::rZIP(.N, mu = 2 * mu, sigma = p)]
zones <- scanstatistics:::powerset_zones(4)
result <- scan_poisson(table, zones, 100)
result
```

score_locations *Score each location over zones and duration.*

Description

For each location, compute the average of the statistic calculated for each space-time window that the location is included in, i.e. average the statistic over both zones and the maximum duration.

Usage

```
score_locations(x)
```

Arguments

x	An object of class `scanstatistic`.

Value

A `data.table` with the following columns:

location The locations (as integers).

total_score For each location, the sum of all window statistics that the location appears in.

n_zones The number of spatial zones that the location appears in.

score The total score divided by the number of zones and the maximum duration.

relative_score The score divided by the maximum score.

Examples

```
# Simple example
set.seed(1)
table <- scanstatistics:::create_table(list(location = 1:4, duration = 1:4),
                                       keys = c("location", "duration"))
table[, mu := 3 * location]
table[, count := rpois(.N, mu)]
table[location %in% c(1, 4) & duration < 3, count := rpois(.N, 2 * mu)]
zones <- scanstatistics:::powerset_zones(4)
result <- scan_poisson(table, zones, 100)
score_locations(result)
```

`top_clusters` *Get the top (non-overlappig) clusters.*

Description

Get the top k space-time clusters according to the statistic calculated for each cluster (the maximum being the scan statistic). The default is to return the spatially non-overlapping clusters, i.e. those that do not have any locations in common.

Usage

```
top_clusters(x, k = 5, overlapping = FALSE)
```

Arguments

`x`	An object of class scanstatistics.
`k`	An integer, the number of clusters to return
`overlapping`	Logical; should the top clusters be allowed to overlap in the spatial dimension? The default is `FALSE`.

Value

A `data.table` with at most k rows, with columns `zone, duration, statistic`.

Examples

```
set.seed(1)
table <- scanstatistics:::create_table(list(location = 1:4, duration = 1:4),
                                       keys = c("location", "duration"))
table[, mu := 3 * location]
table[, count := rpois(.N, mu)]
table[location %in% c(1, 4) & duration < 3, count := rpois(.N, 2 * mu)]
zones <- scanstatistics:::powerset_zones(4)
result <- scan_poisson(table, zones, 0)
top_clusters(result, k = 4, overlapping = FALSE)
```

Index