Package ‘stampr’

October 14, 2022

Type Package

Title Spatial Temporal Analysis of Moving Polygons

Version 0.2

Author Jed Long and Colin Robertson

Maintainer Colin Robertson <crobertson@wlu.ca>

Description Perform spatial temporal analysis of moving polygons; a longstanding analysis problem in Geographic Information Systems. Facilitates directional analysis, shape analysis, and some other simple functionality for examining spatial-temporal patterns of moving polygons.

Depends R (>= 2.10)

License GPL-3

Imports sp, rgeos, raster, rgdal, maptools, deldir, spdep, grDevices, methods

Suggests igraph

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-04-18 22:27:26 UTC

R topics documented:

`stampr-package` *stampr: Spatial Temporal Analysis of Moving Polygons*

Description

The Package `stampr` provides tools for performing spatial temporal analysis of moving polygons. These tools allow the calculation of directional relationships, shape indices, and other basic functionality, such as global change metrics. More details about each of these functions can be found in its help documentation.

Details

`stampr`'s functions utilize the `SpatialPolygonsDataFrame` objects from the package `sp`. Polygon relationships are still understudied in the field of geographic information science, but hopefully `stampr` can provide users with a platform for new developments and applied research looking at interesting geographical phenomena.

Author(s)

Jed Long and Colin Robertson

References

Robertson, C., Nelson, T., Boots, B., and Wulder, M. (2007) STAMP: Spatial-temporal analysis of moving polygons. *Journal of Geographical Systems*, 9:207-227.

`eyeshp` *Hurricane Katrina eye point dataset*

Description

A dataset containing points representing the eye of Hurricane Katrina centroid from 21:00 26-AUG-2005 to 21:00 29-AUG-2005. Polygon contours were extracted from the US NOAA H*Wind product, downloadable from: http://www.aoml.noaa.gov/hrd/data_sub/wind.html

Format

A `SpatialPointsDataFrame` with 33 records of of the location of Hurricane Katrina, every 3 hrs, from 21:00 25-AUG-2005 to 21:00 29-AUG-2005. The date and time of each polygon is recorded in the column `DateTime`.

Details

The eyeshp dataset contains points that were derived from the raw NOAA H*Wind data. The data is included here to provide a point-data comparison to the data in the katrina dataset which is polygon data

Examples

```
library(sp)
data("eyeshp")
plot(eyeshp)
```

---

fire1 *Forest Fire dataset*

Description

A dataset containing fake forest fire polygons representing the movement of the forest fire from T1 (fire1) to T2 (fire2). The data is provided purely for demonstration purposes.

Format

fire1 — a SpatialPolygonsDataFrame with polygons representing the location of forest fire.

Source

Simulated data

Examples

```
library(sp)
data(fire1)
data(fire2)
plot(fire1)
plot(fire2,border=2,add=TRUE)
```

---

fire2 *Forest Fire dataset*

Description

see fire1

Format

fire2 — a SpatialPolygonsDataFrame

glob.change *glob.change*

Description

The function `glob.change` computes a set of three global change metrics for comparison between two polygon sets. These metrics are outlined in Robertson et al. (2007; Table 4).

Usage

```
glob.change(T1, T2)
```

Arguments

`T1`	a `SpatialPolygonsDataFrame` object of polygons from time 1.
`T2`	a `SpatialPolygonsDataFrame` object of polygons from time 2.

Details

`glob.change` computes three change metrics, detailed below, that can be used to quantify changes between two polygon sets:

`NumRatio` – ratio between the number of polygons in `T2` and `T1`;

$$\texttt{NumRatio} = \frac{\#(T1)}{\#(T2)}$$

`AreaRatio` – ratio between the areas of polygons in T2 and T1;

$$\texttt{AreaRatio} = \frac{A(T2)}{A(T1)}$$

`AvgAreaRatio` – ratio between the `AreaRatio` and `NumRatio`;

$$\texttt{AvgAreaRatio} = \frac{\texttt{AreaRatio}}{\texttt{NumRatio}} = \frac{\frac{A(T2)}{A(T1)}}{\frac{\#(T1)}{\#(T2)}}$$

Value

A `list` object with three elements - Results for the `NumRatio`, `AreaRatio`, and `AvgAreaRatio` metrics.

katrina *Hurricane Katrina polygons dataset*

Description

A dataset containing polygons representing the movement of Hurricane Katrina from 21:00 26-AUG-2005 to 21:00 29-AUG-2005. Polygon contours were extracted from the US NOAA H*Wind product, downloadable from: http://www.aoml.noaa.gov/hrd/data_sub/wind.html

Format

A `SpatialPolygonsDataFrame` with 33 records of of the location of Hurricane Katrina, every 3 hrs, from 21:00 25-AUG-2005 to 21:00 29-AUG-2005. The date and time of each polygon is recorded in the column `DateTime`.

Details

The `katrina` dataset contains polygons that were derived from the raw NOAA H*Wind data. The 39 mph isotach (contour of equal wind speed) was used to delineate, as a spatial polygon, the extent of Hurricane Katrina at a given time. Polygons were derived at 3 hr intervals; which means there are 33 different time points in the dataset.

Source

http://www.aoml.noaa.gov/hrd/data_sub/wind.html

References

Powell, M.D., Murillo, S., Dodge, P., Uhlhorn, E., Gamache, J., Cardone, V., Cox, A., Otero, S., Carrasco, N., Annane, B., St. Fleur, R. (2010) Reconstruction of Hurricane Katrina's wind fields for storm surge and wave hindcasting. *Ocean Engineering*, 37, 26-36.

Powell, M.D., Houston, S.H. (1998) The HMD real-time hurricane wind analysis system. *Journal of Wind Engineering and Industrial Aerodynamics*, 77\&78, 53-64.

Examples

```
library(sp)
data(katrina)
plot(katrina, border=1:33)
T1 <- katrina[1,]
plot(T1, col=1, add=TRUE)
```

`mpb` *MPB dataset*

Description

A dataset containing polygons representing the location of mountain pine beetle hotspot polygons in Morice Forest District, British Columbia, Canada.

Format

`mpb` — a `SpatialPolygonsDataFrame` with 711 hotspot polygons that occured over eight years. The temporal indicator is the `TGROUP` column. Another variable `REGION` indicates whether the hotspot was in the northern or southern regions, which experienced mostly independent outbreaks.

Details

These data were derived from helicopter-based GPS surveys during early years of large mountain pine beetle outbreak in Western Canada.

Source

Data obtained from Trisalyn Nelson (ASU)

References

Nelson TA, Boots B, Wulder MA, Carroll AL. Environmental characteristics of mountain pine beetle infestation hot spots. *Journal of Ecosystems and Management*. 2007 Mar 14;8(1).

Examples

```
library(sp)
data(mpb)
plot(mpb,border=2,add=TRUE)
```

`stamp` *Spatial temporal analysis of moving polygons*

Description

This function generates a `SpatialPolygonsDataFrame` that can be used for spatial temporal analysis of moving polygons as described in the paper Robertson et al. (2007).

Usage

```
stamp(T1, T2, dc = 0, direction = FALSE, distance = FALSE, ...)
```

Arguments

`T1`	a `SpatialPolygonsDataFrame` object of polygons from time 1.
`T2`	a `SpatialPolygonsDataFrame` object of polygons from time 2.
`dc`	spatial distance threshold for determining groupings (see **Details**) in appropriate units.
`direction`	logical, whether or not to perform directional analysis. See documentation for `stamp.direction` for further details.
`distance`	logical, whether or not to perform distance analysis. See documentation for `stamp.distance` for further details.
`...`	additional parameters to be passed to functions if `direction`, or `distance` are set to `TRUE`.

Details

The `stamp` function can be used to perform spatial temporal analysis of moving polygons (STAMP) as outlined in the paper by Robertson et al., (2007). Polygon movement "groups" are delineated based on polygon connectedness defined by the distance threshold `dc`. That is, if polygon boundaries (in T1 or T2) are within distance `dc` of one another they will be designated to the same group. STAMP events are reported at four levels of increasing complexity:
LEV1 – disappearance (`DISA`), stable (`STBL`), and generation (`GENA`);
LEV2 – disappearance (`DISA`), contraction (`CONT`), stable (`STBL`), expansion (`EXPN`), and generation (`GENR`);
LEV3 – disappearance (`DISA`), T1 displacement (`DISP1`), convergence (`CONV`), concentration (`CONC`), contraction (`CONT`), stable (`STBL`), expansion (`EXP`), fragmentation (`FRAG`), divergence (`DIV`), T2 displacement (`DISP2`), and generation (`GENR`);
LEV4 – LEV4 is different from other levels. It is used to identify those groups where union (`UNION`), division (`DIVISION`), and both union and division (`BOTH`) events occur. These events occur when there are more than one stable event in a group. Groups with one or no stable events recieve an `NA` value for LEV4.
See Robertson et al. (2007; especially Figure 1) for complete descriptions of all STAMP movement event types.

Note also that there must be a globally unique `ID` column in each data frame passed to the stamp function

Value

This function returns a `SpatialPolygonsDataFrame` with the following data columns:

`ID1`	Polygon ID from T1 polygons; `NA` if it did not exist,
`ID2`	Polygon ID from T2 polygons; `NA` if it did not exist,
`LEV1`	Level 1 STAMP designation,
`LEV2`	Level 2 STAMP designation,
`LEV3`	Level 3 STAMP designation,
`LEV4`	Level 4 STAMP designation,
`GROUP`	Group ID signifying group membership,

`AREA`	Polygon area in appropriate areal units,
`--`	(optional) Additional columns from directional analysis if `direction = TRUE`,
`--`	(optional) Additional columns from distance analysis if `distance = TRUE`,

References

Robertson, C., Nelson, T., Boots, B., and Wulder, M. (2007) STAMP: Spatial-temporal analysis of moving polygons. *Journal of Geographical Systems*, 9:207-227.

See Also

stamp.direction stamp.distance stamp.shape stamp.map stamp.group.summary

---

`stamp.direction` *Perform polygon directional analysis*

Description

`stamp.direction` facilitates polygon directional analysis using a variety of methods.

Usage

```
stamp.direction(stmp, dir.mode = "CentroidAngle", ndir = 4, group = FALSE)
```

Arguments

`stmp`	a `SpatialPolygonsDataFrame` object generated from the `stamp` function.
`dir.mode`	a character item identifying which directional relations method is to be used. See **Details** for information on each individual method.
`ndir`	(optional) parameter identifying the number of directions to be computed. See inidividual method **Details** for appropriate usage.
`group`	(optional) a logical value identifying whether direction should be computed on groups or individual event polygons (only used with `CentroidAngle` method).

Details

The `stamp.direction` function can be used to facilitate directional analysis on output `stamp.obj` objects from function `stamp`. Currently, four directional analysis methods are available:

- `"CentroidAngle"` – The centroid angle is simply the angle between the centroids of two polygons. The centroid angle method is computed on STAMP objects by first grouping all T1 polygons (by STAMP group) and computing their centroid. Then, the angle from each T1 group centroid, to the centroid of each STAMP event within the group is calculated. Centroid angles are recorded in degrees, with North having a value of 0, East 90, and so on. `"CentroidAngle"` ignores the `ndir` parameter.

- "ConeModel" – The cone model method calculates areas of STAMP event polygons within cones radiating from the centroid of the origin polygon. The cone model method first computes the centroid of all T1 polygons in a STAMP grouping. It then computes ndir equally spaced cones radiating outward from the T1 centroid. The first cone is always centered on North, but there can be any number of cones. The area of each STAMP event, in each cone (specifying direction), is then calculated. See Peuquet and Zhang (1987) for more detailed information
- "MBRModel" – The minimum bounding rectangle (MBR) method first computes the MBR for all T1 events in a STAMP grouping. Then the lines of four edges of the MBR are extended outwards to infinity creating sections for the eight cardinal directions around the MBR, along with the MBR itself. The area of each stamp event within each of the nine sections is then computed. See Skiadopoulos et al. (2005) for more detailed information. "MBRModel" ignores the ndir parameter.
- "ModConeModel" – The modified cone model first computes the centroid of the T1 event that includes a stable event type. Then ndir = 4 or 8 cones are created outward from this centroid to the minimum bounding rectangle of the entire grouping. As described by Robertson et al. (2007) this approach is more accomodating to polygon groups that are irregular in size or shape. If there is more than 1 stable event (as flagged by the stamp.obj LEV4 column, the Voronoi segregation method defined by Robertson et al. (2007) is employed. The modified cone model method first computes the centroid of all T1 polygons in a STAMP grouping. It then computes the bounding box of ALL events in a STAMP grouping. Then, ndir=4 or 8 cones are computed. In the case of ndir=4, cones radiate from the T1 centroid to the four corners of the bounding box. The result of the modified cone model method is that the cones are not equally spaced, but tailored to the individual STAMP groupings shape. See Robertson et al. (2007) for more detailed information.

Value

Appends the input stamp object with appropriate columns for the directional analysis chosen, if dir.mode is:

"CentroidAngle"
: A single column with centroid angle results, in degrees (North = 0 degrees). If group=TRUE then values are identical for all event polygons in the group.

"ConeModel"
: ndir new columns with the area of the STAMP event in each direction, named appropriately (e.g., as DIR45, where 45 refers to the mid-point of that directional cone).

"MBRModel"
: 9 new columns with the area of the STAMP event in each direction, named appropriately as "SW","S","SE","W","SAME","E","NW","N","NE".

"ModConeModel"
: ndir new columns with the area of the STAMP event in each direction, named appropriately as, for example, "N","E","S","W" with ndir=4.

Note: STAMP events that are singular (i.e., only 1 polygon in the group) will have NA's from directional analysis.

References

Robertson, C., Nelson, T., Boots, B., and Wulder, M. (2007) STAMP: Spatial-temporal analysis of moving polygons. *Journal of Geographical Systems*, 9:207-227.

Peuquet, D., Zhang, C.X. (1987) An algorithm to determine the directional relationship between arbitrarily-shaped polygons in the plane. *Pattern Recognition*, 20:65-74.

Skiadopoulos, S. Giannoukos, C., Sarkas, N., Vassiliadis, P., Sellis, T., and Koubarakis, M. (2005) Computing and managing directional relations. *IEEE Transactions on Knowledge and Data Engineering*, 17:1610-1623.

See Also

stamp, stamp.distance, stamp.shape

`stamp.distance` *stamp.distance*

Description

The function `stamp.distance` can be used to compute various measures of distance between polygon events and groups. In turn, distance measurements can be used to estimate the velocity of polygon movement.

Usage

```
stamp.distance(stmp, dist.mode = "Centroid", group = FALSE)
```

Arguments

`stmp`	a `SpatialPolygonsDataFrame` object generated from the `stamp` function.
`dist.mode`	Character determining the emethod by which polygon distances are computed. If `"Centroid"` then the centroid distance is calculated, if `"Hausdorff"` then the discrete Hausdorff distance is calculated; see `Details`.
`group`	logical indicating whether distances should be computed from the T1 polygon to each individual stamp event (`group = FALSE` – the default), or whether T2 polygons should combined (through a spatial union) in order to compute the measure of distance for each stamp group (`group = TRUE`)

Details

`stamp.distance` computes distance between polygon sets based on either centroid or Hausdorff distance calculations. Centroid distance is simply the distance from the centroid of all T1 polygons (combined) to each stamp event (`group = FALSE`), or to the union of all T2 polygons within a group (`group = TRUE`), in the second case, all events within a group are given an identical distance value.

The Hausdorff distance calculation uses the discrete version of the Hausdorff distance, as programmed in the `rgeos` function `gDistance`. A value of `densifyFrac = 1` is used to increase the precision of this measurement – see `help(gDistance)`. The returned distance is then the Hausdorff distance of all T1 polygons (combined) to each stamp event (`group = FALSE`), or to the union of all T2 polygons within a group (`group = TRUE`), in the second case, all events within a group are given an identical distance value.

Value

Appropriately named columns (e.g., CENDIST or HAUSDIST) in the stamp SpatialPolygonsDataFrame object.

References

Hausdorff Distance: http://en.wikipedia.org/wiki/Hausdorff_distance

See Also

stamp stamp.direction stamp.shape gDistance

---

stamp.group.summary *Compile stamp summary statistics by group*

Description

The function stamp.group.summary compiles summary statistics for each STAMP grouping. Specifically, it computes the area of each STAMP event type (e.g., generation, expansion, etc.) within each grouping. It also computes the number of events belonging to each event type.

Usage

```
stamp.group.summary(stmp, area = TRUE, count = TRUE)
```

Arguments

stmp	a SpatialPolygonsDataFrame generated from the stamp function.
area	logical, whether or not to compute the STAMP event areas.
count	logical, whether or not to compute the count of STAMP evets within each group.

Details

stamp.group.summary computes area and count summary statistics of STAMP output. Note that if both area and count are set to FALSE, stamp.group.summary returns a data.frame with just the group IDs as the only column.

Value

A data.frame where rows are stamp groups and columns correspond to the STAMP event types (ID, areas, and counts).

stamp.map *Mapping (plotting) functionality for `stamp` output*

Description

This function maps STAMP output for visual assessment of STAMP events and groupings. Choice of which aspect of the stamp output to be visualized is controlled by passing the column name to the `stamp.map` function.

Usage

```
stamp.map(stmp, by = "LEV1", ...)
```

Arguments

`stmp`	output from the `stamp` function, i.e., a (`SpatialPolygonsDataFrame`).
`by`	tells the function which attribute to visualize, one of `"LEV1"`, `"LEV2"`, `"LEV3"`, `"LEV4"`, or `"GROUP"`
`...`	additional parameters to be passed to the plot function

Details

The `stamp.map` function can be used to visualize any of the stamp event designation levels (e.g., `"LEV1"`, `"LEV2"`, `"LEV3"`, `"LEV4"`, or the STAMP groupings (based off of parameter `dc` in the `stamp` function).

Value

`stamp.map` returns a map of the `stamp` output using the `spplot` functionality. It implements a pre-defined coloring scheme.

See Also

stamp data("fire1") data("fire2") fire1$ID <- 1:nrow(fire1) fire2$ID <- (max(fire1$ID)+1):(max(fire1$ID) + nrow(fire2)) ch <- stamp(fire1, fire2, dc=1, direction=FALSE, distance=FALSE) stamp.map(ch, "LEV1") stamp.map(ch, "LEV2") stamp.map(ch, "LEV3") stamp.map(ch, "LEV4") `stamp.multichange` *run stamp function for multiple years of polygons at once*

Description

The function `stamp.multichange` is a wrapper function that makes multiple calls to the stamp function to ease spatial-temporal analysis of multiple years of polygon data

Usage

```
stamp.multichange(polys, changeByRow = TRUE, changeByField = FALSE,
  changeField = "", ...)
```

Arguments

`polys`	a `SpatialPolygonsDataFrame` wiht 2+ years of data to run through the `stamp` function.
`changeByRow`	logical, whether or not each time period is a separate unique row of data (e.g., as per the `katrina` data)
`changeByField`	logical, whether or not time period data is given by a specific field. If this is TRUE, changeByRow should be false
`changeField`	string, name of the field which contains time period if changeByField is TRUE
`...`	list of paramater values to provide to the `stamp` function

Details

`stamp.multichange` is a simple wrapper function for the `stamp` function. The two options for data structure are those in the `katrina` data, where each time period is a row, and rows are time-ordered, and the structure of the `mpb` data, where time period is specified by a column. Time periods should be ordered from 1 through T.

Value

A `SpatialPolygonsDataFrame` which includes all outputs from the calls to the `stamp` function. If there are T time periods, there will be T-1 time periods in the resulting `SpatialPolygonsDataFrame` object.

Examples

```
library(sp)
data("katrina")
katrina$ID <- katrina$Id
ch <- stamp.multichange(katrina, changeByRow = TRUE, dc = 0, distance = TRUE, direction = FALSE)
STGroup <- stamp.stgroup.summary(ch)
head(STGroup)
```

stamp.shape — *Compute shape indices on `stamp` output*

Description

This function computes a suite of shape complexity metrics on STAMP polygons facilitating shape analysis.

Usage

```
stamp.shape(T1, T2, stmp, index = "PAR")
```

Arguments

`T1`	a `SpatialPolygons` object of polygons from time 1.
`T2`	a `SpatialPolygons` object of polygons from time 2.
`stmp`	output `SpatialPolygonsDataFrame` generated from the `stamp` function.
`index`	a character item identifying which shape metric is to be computed. See **Details**.

Details

The `stamp.shape` function can be used to perform polygon shape analysis on output polygons from function `stamp`. Shape indices are computed on each output polygon. Five shape indices are available:
`"PER"` – Shape perimeter, in appropriate units.
`"PAR"` – Perimeter-area ratio, in appropriate units;

$$\texttt{PAR} = \frac{p}{a}$$

`"FRAC"` – Fractal dimension (Mandelbrot 1977, Lovejoy 1982);

$$\texttt{FRAC} = \frac{2\log(p)}{\log(a)}$$

`"SHPI"` – Shape index (Patton 1975);

$$\texttt{SHPI} = \frac{p}{2 * \sqrt{\pi * a}}$$

`"LIN"` – Linearity index (Baker and Cai 1992);

$$\texttt{LIN} = 1 - \frac{a}{a_{circ}}$$

Where a is polygon area, p is polygon perimeter, and a_{circ} is the area of the circumscribing (encompassing) circle of a polygon.

Some Notes:
PER is simply the length of the perimeter, and is not an overly useful measure of shape, but may be useful in direct comparisons. PAR > 0, without limit with larger values sugesting more complex, irregular shapes. The range of FRAC is [1, 2]. FRAC approaches 1 for very simple shapes (squares, circles, etc.) and approaches 2 for complex, convoluted shapes. SHPI > 1 without limit, as SHPI increase, the complexity of the shape increases. The range of LIN is [0, 1]. A perfect circle will have a LIN of 0, while more linear shapes will approach 1.

The indices PAR, FRAC, SHPI, and LIN are all essentially measures of shape complexity. LIN is unique in that it tries to focus on the linearity of the shape by comparing the area to a circle. LIN is however, less useful with STAMP events containing multiple polygons, as the calculation for the circumscribing circle will include all polygon objects within the group and artificially increase the LIN scores.

Value

A `DataFrame` with four columns:
`GROUP` – STAMP polygon groups from the `stamp` function. `T1.INDEX` – shape index value for T1 polygons for each group. `INDEX` is replaced by name of index. `T2.INDEX` – shape index value for T2 polygons for each group. `INDEX` is replaced by name of index. `d.INDEX` – change (t2 - t1) in shape value for each group. `INDEX` is replaced by name of index.

References

Baker, W.L. and Cai, Y. (1992) The r.le programs for multiscale analysis of landscape structure using the GRASS geographical information system. *Landscape Ecology*, 7(4):291-302.

Lovejoy, S. (1982) Area-perimeter relation for rain and cloud areas. *Science*, 216(4542):185-187.

Mandlebrot, B.B. (1977) Fractals, Form, Chance and Dimension. W.H Freeman and Co., New York.

Patton, D.R. (1977) A diversity index for quantifying habitat "edge". *Wildlife Society Bulletin*, 3:171-173.

See Also

stamp

Examples

```
library(sp)
library(rgeos)
library(raster)
data("fire1")
data("fire2")
#set globally unique ID column required for stamp function
fire1$ID <- 1:nrow(fire1)
#set globally unique ID column required for stamp function
```

```
fire2$ID <- (max(fire1$ID)+1):(max(fire1$ID) + nrow(fire2))
ch <- stamp(fire1, fire2, dc=1, direction=FALSE, distance=FALSE)
ch.sh <- stamp.shape(T1 = fire1, T2 = fire2, stmp = ch, index = 'LIN')
```

stamp.stgroup.summary *Compile stamp summary statistics by space-time group*

Description

The function `stamp.stgroup.summary` compiles summary statistics for each STAMP grouping. Specifically, it computes the area of each STAMP event type (e.g., generation, expansion, etc.) within each grouping. It also computes the number of events belonging to each event type.

Usage

```
stamp.stgroup.summary(stmp, area = TRUE, count = TRUE)
```

Arguments

`stmp`	a `SpatialPolygonsDataFrame` generated from the `stamp` function.
`area`	logical, whether or not to compute the STAMP event areas.
`count`	logical, whether or not to compute the count of STAMP evets within each group.

Details

`stamp.group.summary` computes area and count summary statistics of STAMP output. Note that if both `area` and `count` are set to `FALSE`, `stamp.group.summary` returns a `data.frame` with just the stgroup IDs as the only column.

Value

A `data.frame` where rows are stamp groups and columns correspond to the STAMP event types (ID, areas, and counts).

Examples

```
library(sp)
data("katrina")
katrina$ID <- katrina$Id
ch <- stamp.multichange(katrina, changeByRow = TRUE, dc = 0, distance = TRUE, direction = FALSE)
STGroup <- stamp.stgroup.summary(ch)
head(STGroup)
```

Index