Package: tidyrgeoda (via r-universe)

July 3, 2024

Title A tidy interface for rgeoda

Version 0.1.1

Description An interface for 'rgeoda' to integrate with 'sf' objects and the 'tidyverse'.

License GPL-3

URL https://github.com/SpatLyu/tidyrgeoda, https://spatlyu.github.io/tidyrgeoda/

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports sf, rgeoda, magrittr, dplyr, tibble, ggplot2, stringr

Suggests tmap, rlang, knitr, rmarkdown

Depends R (>= 4.1.0)

VignetteBuilder knitr

LazyData true

Repository https://spatlyu.r-universe.dev

RemoteUrl https://github.com/SpatLyu/tidyrgeoda

RemoteRef HEAD

RemoteSha 21836bc55d9b9f4c741aa929528873b8141f535c

Contents

`gzma` *Guangzhou Metropolitan Social Space Quality Score Data*

Description

Data of Social Space Quality Score in Guangzhou Metropolitan Areas of China (2010). The actual representation of each column is as follows: `SName_EN` street name; `DName_EN` district name; `SSQ_Score` total score of social space quality; `PS_Score` population stability score; `EL_Score` educational level score; `OH_Score` occupational hierarchy score; `IL_Score` income level score; `FPOP_Pro` proportion of foreign population; `TenantsPro` proportion of tenants; `NoSchPro` proportion of no schooling; `PSchPRO` proportion of primary school education; `JHSchPro` proportion of junior high school education; `HSchDipPro` proportion of high school diploma; `CDegreePro` proportion of college degree; `UnderG_Pro` proportion of undergraduate; `PostG_Pro` proportion of postgraduate; `RPSOPMOPro` proportion of responsible persons of state organs, party and mass organizations and institutions; `PCE_Pro` proportion of person in charge of enterprise; `ProTechPro` proportion of professional and technical personnel; `ClerkPro` proportion of clerk; `BusSer_Pro` proportion of business and service personnel; `AFAFP_Pro` proportion of agriculture, forestry, animal

husbandry and fishery personnel; OPTE_Pro proportion of operators of production and transportation equipment; UnemPeoPro proportion of households with unemployed people; B100_Pro proportion of households with below 100 yuan; 100_200Pro proportion of households with 100-200yuan; 200_500Pro proportion of households with 200-500yuan; 500_1000P proportion of households with 500-1000yuan; 1000_1500P proportion of households with 1000-1500yuan; 1500_2000P proportion of households with 1500-2000yuan; 2000_3000P proportion of households with 2000-3000yuan; A3000_Pro proportion of households with above 3000yuan; The subdistrict boundary is drawn by with reference to the Atlas of Community Network ResponsibilityDistrict of Urban Management Division in Guangzhou.

Usage

```
gzma
```

Format

gzma: An sf tibble of social space quality score in guangzhou metropolitan areas(2010) with 118 rows and 32 variables, where the last column is geometry.

References

WANG Yang, ZHANG Hong'ou, YE Yuyao, WU Qitao, JIN Lixia. Comprehensive Evaluation and Distribution Pattern of Social Space Quality in Guangzhou, China. Tropical Geography.

---

read_geoda *Read A Geoda File(.gal,.gwt,.swm)*

Description

Create a spatial weights object from a geoda file

Usage

```
read_geoda(file_path, id_vec = NULL)
```

Arguments

file_path	The file paht of the geoda file.
id_vec	(optional),the id_vec is the id values used in the geoda file.

Value

A weights object

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

scale_fill_lisa *LISA Fill Scales*

Description

Provide ggplot2 fill scales like geoda software.Now it achieve by using ?ggplot2::scale_fill_manual().Another achieve can see https://stackoverflow.com/questions/43440068/ggplot2-fix-colors-to-factor-l.

Usage

```
scale_fill_lisa(name = "LISA", ...)
```

Arguments

name	The name of the LISA fill scales legend,default is LISA
...	Adjust other legend details for the LISA fill scales, like labels to adjust the appearance of legend, see ?ggplot2::scale_fill_manual().

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
library(ggplot2)
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = read_sf(guerry_path)
guerry |>
 dplyr::mutate(lisa = st_local_moran(guerry,'Crm_prs')) |>
 dplyr::select(lisa) |>
 ggplot() +
 geom_sf(aes(fill = lisa),lwd = .1,color = 'grey') +
 scale_fill_lisa()
```

st_azp_greedy *A greedy algorithm to solve the AZP problem*

Description

A wrapper function for rgeoda::azp_greedy().The automatic zoning procedure (AZP) was initially outlined in Openshaw (1977) as a way to address some of the consequences of the modifiable areal unit problem (MAUP). In essence, it consists of a heuristic to find the best set of combinations of contiguous spatial units into p regions, minimizing the within sum of squares as a criterion of homogeneity. The number of regions needs to be specified beforehand.

Usage

```
st_azp_greedy(
  sfj,
  varcol,
  k,
  wt = NULL,
  boundvar = NULL,
  min_bound = 0,
  inits = 0,
  initial_regions = vector("numeric"),
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  rdist = numeric()
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`k`	The number of clusters.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`boundvar`	(optional) A data frame / tibble with selected bound variable.
`min_bound`	(optional) A minimum bound value that applies to all clusters.
`inits`	(optional) The number of construction re-runs, which is for ARiSeL "automatic regionalization with initial seed location".
`initial_regions`	(optional) The initial regions that the local search starts with. Default is empty. means the local search starts with a random process to "grow" clusters.
`scale_method`	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
`distance_method`	(optional) The distance method used to compute the distance betwen observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan"
`seed`	(optional) The seed for random number generator. Defaults to 123456789.
`rdist`	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_azp_greedy(guerry,c('Crm_prs','Crm_prp','Litercy',
'Donatns','Infants','Suicids'),5)
guerry_clusters
```

---

st_azp_sa — *A simulated annealing algorithm to solve the AZP problem*

Description

A wrapper function for `rgeoda::azp_sa()`.The automatic zoning procedure (AZP) was initially outlined in Openshaw (1977) as a way to address some of the consequences of the modifiable areal unit problem (MAUP). In essence, it consists of a heuristic to find the best set of combinations of contiguous spatial units into p regions, minimizing the within sum of squares as a criterion of homogeneity. The number of regions needs to be specified beforehand.

Usage

```
st_azp_sa(
  sfj,
  varcol,
  k,
  wt = NULL,
  boundvar = NULL,
  cooling_rate = 0.85,
  sa_maxit = 1,
  min_bound = 0,
  inits = 0,
  initial_regions = vector("numeric"),
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  rdist = numeric()
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`k`	The number of clusters.

wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
boundvar	(optional) A data frame / tibble with selected bound variable.
cooling_rate	(optional) The cooling rate of a simulated annealing algorithm. Defaults to 0.85.
sa_maxit	(optional) The number of iterations of simulated annealing. Defaults to 1.
min_bound	(optional) A minimum bound value that applies to all clusters.
inits	(optional) The number of construction re-runs, which is for ARiSeL "automatic regionalization with initial seed location".
initial_regions	(optional) The initial regions that the local search starts with. Default is empty. means the local search starts with a random process to "grow" clusters.
scale_method	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
distance_method	(optional) The distance method used to compute the distance betwen observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan"
seed	(optional) The seed for random number generator. Defaults to 123456789.
rdist	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_azp_sa(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns',
'Infants','Suicids'),5)
guerry_clusters
```

---

st_azp_tabu *A tabu algorithm to solve the AZP problem*

Description

A wrapper function for `rgeoda::azp_tabu()`.The automatic zoning procedure (AZP) was initially outlined in Openshaw (1977) as a way to address some of the consequences of the modifiable areal unit problem (MAUP). In essence, it consists of a heuristic to find the best set of combinations of contiguous spatial units into p regions, minimizing the within sum of squares as a criterion of homogeneity. The number of regions needs to be specified beforehand.

Usage

```
st_azp_tabu(
  sfj,
  varcol,
  k,
  wt = NULL,
  boundvar = NULL,
  tabu_length = 10,
  conv_tabu = 10,
  min_bound = 0,
  inits = 0,
  initial_regions = vector("numeric"),
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  rdist = numeric()
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`k`	The number of clusters.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`boundvar`	(optional) A data frame / tibble with selected bound variable.
`tabu_length`	(optional) The length of a tabu search heuristic of tabu algorithm. Defaults to 10.
`conv_tabu`	(optional): The number of non-improving moves. Defaults to 10.
`min_bound`	(optional) A minimum bound value that applies to all clusters.
`inits`	(optional) The number of construction re-runs, which is for ARiSeL "automatic regionalization with initial seed location".
`initial_regions`	(optional) The initial regions that the local search starts with. Default is empty. means the local search starts with a random process to "grow" clusters.
`scale_method`	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
`distance_method`	(optional) The distance method used to compute the distance betwen observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan"
`seed`	(optional) The seed for random number generator. Defaults to 123456789.
`rdist`	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_azp_tabu(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns',
'Infants','Suicids'),5)
guerry_clusters
```

---

st_breaks *Univariate Spatial Stratification*

Description

Univariate Spatial Stratification by invoking rgeoda's *_breaks function.

Usage

```
st_breaks(sfj, varcol, break_method = "stddev", k = 6)
```

Arguments

sfj	An sf, tibble or data.frame object
varcol	The variables selected to run univariate spatial stratification.
break_method	(optional) Which has to be one of "stddev"(default), "hinge15", "hinge30", "percentile", "natural", "quantile". When the break_method is "natural" or "quantile", you can assign the k argument to have how many breaks, other methods will have 6 groups.
k	(optional)A numeric value indicates how many breaks,default is 6.

Value

A vector of numeric values of computed breaks

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
st_breaks(guerry,'Crm_prs',break_method = "quantile", k = 5)
```

---

st_contiguity_weights *Contiguity Spatial Weights*

Description

Create a contiguity spatial weights with options of "queen", "order", "include lower order" and "precision threshold"

Usage

```
st_contiguity_weights(
  sfj,
  queen = TRUE,
  order = 1,
  include_lower_order = FALSE,
  precision_threshold = 0
)
```

Arguments

sfj	An sf (simple feature) object.
queen	(Optional) TRUE (default) or FALSE, TRUE implements Queen Contiguity and FALSE implements Rook Contiguity.
order	(Optional) Order of contiguity, default is 1.
include_lower_order	(Optional) Whether or not the lower order neighbors should be included in the weights structure,default is False.
precision_threshold	(Optional) The precision of the underlying shape file is insufficient to allow for an exact match of coordinates to determine which polygons are neighbors,default is 0.

Value

An instance of rgeoda Weight-class.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
queenw = st_contiguity_weights(guerry,queen = TRUE)
```

---

st_distance_weights *Distance-based Spatial Weights*

Description

Create a distance-based weights

Usage

```
st_distance_weights(
  sfj,
  unit = "km",
  dist_thres = NULL,
  power = 1,
  is_inverse = FALSE
)
```

Arguments

sfj	An sf (simple feature) object.
unit	(optional) The unit for calculating spatial distance, can be 'km'(default) or 'mile'.
dist_thres	(optional) A positive numeric value of distance threshold.
power	(optional) The power (or exponent) of a number indicates how many times to use the number in a multiplication.Default is 1.
is_inverse	(optional) FALSE (default) or TRUE, apply inverse on distance value.

Value

An instance of rgeoda Weight-class.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_distance_weights(guerry)
```

st_kernel_knn_weights *K-NN Kernel Spatial Weights*

Description

Create a kernel weights by specifying k-nearest neighbors and a kernel method

Usage

```
st_kernel_knn_weights(
  sfj,
  k,
  kernel = "gaussian",
  power = 1,
  adaptive_bandwidth = TRUE,
  use_kernel_diagonals = FALSE,
  is_inverse = FALSE,
  unit = "km"
)
```

Arguments

sfj	An sf (simple feature) object.
k	A positive integer number for k-nearest neighbors.
kernel	(optional) A string value, which has to be one of 'triangular', 'uniform', 'epanechnikov', 'quartic', 'gaussian'(default).
power	(optional) The power (or exponent) of a number indicates how many times to use the number in a multiplication.Default is 1.
adaptive_bandwidth	(optional) TRUE (default) or FALSE: TRUE use adaptive bandwidth calculated using distance of k-nearest neithbors, FALSE use max distance of all observation to their k-nearest neighbors.
use_kernel_diagonals	(optional) FALSE (default) or TRUE, apply kernel on the diagonal of weights matrix.
is_inverse	(optional) FALSE (default) or TRUE, apply inverse on distance value.
unit	(optional) The unit for calculating spatial distance, can be 'km'(default) or 'mile'.

Value

An instance of rgeoda Weight-class.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_kernel_knn_weights(guerry,6)
```

---

st_kernel_weights *Distance-based Kernel Spatial Weights*

Description

Create a kernel weights by specifying a bandwidth and a kernel method

Usage

```
st_kernel_weights(
  sfj,
  kernel = "gaussian",
  bandwidth = NULL,
  power = 1,
  use_kernel_diagonals = FALSE,
  is_inverse = FALSE,
  unit = "km"
)
```

Arguments

sfj	An sf (simple feature) object.
kernel	(optional) A string value, which has to be one of 'triangular', 'uniform', 'epanechnikov', 'quartic', 'gaussian'(default).
bandwidth	(optional) A positive numeric value of bandwidth.
power	(optional) The power (or exponent) of a number indicates how many times to use the number in a multiplication.Default is 1.
use_kernel_diagonals	(optional) FALSE (default) or TRUE, apply kernel on the diagonal of weights matrix.
is_inverse	(optional) FALSE (default) or TRUE, apply inverse on distance value.
unit	(optional) The unit for calculating spatial distance, can be 'km'(default) or 'mile'.

Value

An instance of rgeoda Weight-class.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_kernel_weights(guerry)
```

---

st_knn_weights *K-Nearest Neighbors-based Spatial Weights*

Description

Create a k-nearest neighbors based spatial weights

Usage

```
st_knn_weights(sfj, k, power = 1, is_inverse = FALSE, unit = "km")
```

Arguments

sfj	An sf (simple feature) object.
k	A positive integer number for k-nearest neighbors.
power	(optional) The power (or exponent) of a number indicates how many times to use the number in a multiplication.Default is 1.
is_inverse	(optional) FALSE (default) or TRUE, apply inverse on distance value.
unit	(optional) The unit for calculating spatial distance, can be 'km'(default) or 'mile'.

Value

An instance of rgeoda Weight-class.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_knn_weights(guerry,3)
```

st_lag *Spatial Lag*

Description

Compute the spatial lag for idx-th observation using selected variable and spatial weights matrix

Usage

```
st_lag(sfj, varcol, wt = NULL)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.

Value

A numeric vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_lag(guerry,'Pop1831')
```

st_lnmt *Local Neighbor Match Test*

Description

The local neighbor match test is to assess the extent of overlap between k-nearest neighbors in geographical space and k-nearest neighbors in multi-attribute space.

Usage

```
st_lnmt(
  sfj,
  varcol,
  k,
  unit = "km",
  scale_method = "standardize",
  distance_method = "euclidean",
  power = 1,
  is_inverse = FALSE
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variables selected to run local neighbor match test.
k	A positive integer number for k-nearest neighbors searching.
unit	(optional) The unit for calculating spatial distance, can be 'km'(default) or 'mile'.
scale_method	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
distance_method	(optional) The type of distance metrics used to measure the distance between input data. Options are 'euclidean', 'manhattan'. Default is 'euclidean'.
power	(optional) The power (or exponent) of a number says how many times to use the number in a multiplication.
is_inverse	(optional) FALSE (default) or TRUE, apply inverse on distance value.

Value

A tibble with two columns "Cardinality" and "Probability".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata","Guerry.shp",package = "rgeoda"))
st_lnmt(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns',
'Infants','Suicids'),6)
```

st_local_bijoincount *Bivariate Local Join Count Statistics*

Description

Function to apply local Bivariate Join Count statistics

Usage

```
st_local_bijoincount(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`permutations`	(optional) The number of permutations for the LISA computation.
`permutation_method`	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
`significance_cutoff`	(optional) A cutoff value for significance p-values to filter not-significant clusters.
`cpu_threads`	(optional) The number of cpu threads used for parallel LISA computation.
`seed`	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(magrittr)
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path) %>% dplyr::mutate(InvCrm = 1 - TopCrm)
st_local_bijoincount(guerry,c("TopCrm", "InvCrm"))
```

---

st_local_bimoran *Bivariate Local Moran Statistics*

Description

Function to apply bivariate local Moran statistics

Usage

```
st_local_bimoran(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`permutations`	(optional) The number of permutations for the LISA computation.
`permutation_method`	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
`significance_cutoff`	(optional) A cutoff value for significance p-values to filter not-significant clusters.
`cpu_threads`	(optional) The number of cpu threads used for parallel LISA computation.
`seed`	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_bimoran(guerry,c('Crm_prs','Litercy'))
```

---

st_local_g *Local Getis-Ord's G Statistics*

Description

Function to apply Getis-Ord's local G statistics

Usage

```
st_local_g(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`permutations`	(optional) The number of permutations for the LISA computation.
`permutation_method`	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
`significance_cutoff`	(optional) A cutoff value for significance p-values to filter not-significant clusters.
`cpu_threads`	(optional) The number of cpu threads used for parallel LISA computation.
`seed`	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_g(guerry,'Crm_prp')
```

---

st_local_geary *Local Geary Statistics*

Description

Function to apply local Geary statistics

Usage

```
st_local_geary(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
permutations	(optional) The number of permutations for the LISA computation.
permutation_method	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
significance_cutoff	(optional) A cutoff value for significance p-values to filter not-significant clusters.
cpu_threads	(optional) The number of cpu threads used for parallel LISA computation.
seed	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_geary(guerry,'Crm_prp')
```

---

st_local_gstar *Local Getis-Ord's G* Statistics*

Description

Function to apply Getis-Ord's local G*statistics

Usage

```
st_local_gstar(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
permutations	(optional) The number of permutations for the LISA computation.
permutation_method	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
significance_cutoff	(optional) A cutoff value for significance p-values to filter not-significant clusters.
cpu_threads	(optional) The number of cpu threads used for parallel LISA computation.
seed	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_gstar(guerry,'Crm_prp')
```

---

st_local_joincount *Local Join Count Statistics*

Description

Function to apply local Join Count statistics

Usage

```
st_local_joincount(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
permutations	(optional) The number of permutations for the LISA computation.
permutation_method	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
significance_cutoff	(optional) A cutoff value for significance p-values to filter not-significant clusters.
cpu_threads	(optional) The number of cpu threads used for parallel LISA computation.
seed	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_joincount(guerry,'Crm_prp')
```

---

st_local_moran *Local Moran Statistics*

Description

Function to apply local Moran statistics

Usage

```
st_local_moran(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
permutations	(optional) The number of permutations for the LISA computation.
permutation_method	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
significance_cutoff	(optional) A cutoff value for significance p-values to filter not-significant clusters.
cpu_threads	(optional) The number of cpu threads used for parallel LISA computation.
seed	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_moran(guerry,'Crm_prp')
```

---

st_local_moran_eb *Local Moran with Empirical Bayes(EB) Rate*

Description

Function to apply local Moran with EB Rate statistics. The EB rate is first computed from "event" and "base" variables, and then used in local moran statistics.

Usage

```
st_local_moran_eb(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`permutations`	(optional) The number of permutations for the LISA computation.
`permutation_method`	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
`significance_cutoff`	(optional) A cutoff value for significance p-values to filter not-significant clusters.

cpu_threads	(optional) The number of cpu threads used for parallel LISA computation.
seed	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
## Not run:
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_moran_eb(guerry,c("hr60", "po60"))

## End(Not run)
```

---

st_local_multigeary *Local Multivariate Geary Statistics*

Description

Function to apply local Multivariate Geary statistics

Usage

```
st_local_multigeary(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variables selected to calculate spatial lag, which is a character vector.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
permutations	(optional) The number of permutations for the LISA computation.

`permutation_method`
: (optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.

`significance_cutoff`
: (optional) A cutoff value for significance p-values to filter not-significant clusters.

`cpu_threads`
: (optional) The number of cpu threads used for parallel LISA computation.

`seed`
: (optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_multigeary(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns','Infants',
'Suicids'))
```

---

st_local_multijoincount

(Multivariate) Colocation Local Join Count Statistics

Description

Function to apply (multivariate) colocation local Join Count statistics

Usage

```
st_local_multijoincount(
  sfj,
  varcol,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`permutations`	(optional) The number of permutations for the LISA computation.
`permutation_method`	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
`significance_cutoff`	(optional) A cutoff value for significance p-values to filter not-significant clusters.
`cpu_threads`	(optional) The number of cpu threads used for parallel LISA computation.
`seed`	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_multijoincount(guerry,c('TopWealth','TopWealth', 'TopLit'))
```

---

st_local_multiquantilelisa

Multivariate Quantile LISA Statistics

Description

Function to apply multivariate quantile LISA statistics

Usage

```
st_local_multiquantilelisa(
  sfj,
  varcol,
  k,
  q,
  wt = NULL,
```

```
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
k	A value indicates the number of quantiles. Value range e.g. [1, 10].
q	A value indicates which quantile or interval used in local join count statistics. Value stars from 1.
wt	(optional) The spatial weights object,which can use st_weights() to construct,default is constructed by st_weights(sfj,'contiguity').
permutations	(optional) The number of permutations for the LISA computation.
permutation_method	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
significance_cutoff	(optional) A cutoff value for significance p-values to filter not-significant clusters.
cpu_threads	(optional) The number of cpu threads used for parallel LISA computation.
seed	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_multiquantilelisa(guerry,c("Crm_prp", "Litercy"),c(4,4), c(1,1))
```

st_local_quantilelisa *Quantile LISA Statistics*

Description

Function to apply quantile LISA statistics

Usage

```
st_local_quantilelisa(
  sfj,
  varcol,
  k,
  q,
  wt = NULL,
  permutations = 999,
  permutation_method = "complete",
  significance_cutoff = 0.05,
  cpu_threads = 6,
  seed = 123456789
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
k	A value indicates the number of quantiles. Value range e.g. `[1, 10]`.
q	A value indicates which quantile or interval used in local join count statistics. Value stars from 1.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
permutations	(optional) The number of permutations for the LISA computation.
permutation_method	(optional) The permutation method used for the LISA computation. Options are 'complete', 'lookup'. Default is 'complete'.
significance_cutoff	(optional) A cutoff value for significance p-values to filter not-significant clusters.
cpu_threads	(optional) The number of cpu threads used for parallel LISA computation.
seed	(optional) The seed for random number generator.

Value

A factor vector.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_local_quantilelisa(guerry,"Crm_prs",k = 4, q = 1)
```

---

st_maxp_greedy *A greedy algorithm to solve the max-p-region problem*

Description

A wrapper function for `rgeoda::maxp_greedy()`.The max-p-region problem is a special case of constrained clustering where a finite number of geographical areas are aggregated into the maximum number of regions (max-p-regions), such that each region is geographically connected and the clusters could maximize internal homogeneity.

Usage

```
st_maxp_greedy(
  sfj,
  varcol,
  wt = NULL,
  boundvar,
  min_bound,
  iterations = 99,
  initial_regions = vector("numeric"),
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  cpu_threads = 6,
  rdist = numeric()
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`boundvar`	A numeric vector of selected bounding variable.
`min_bound`	A minimum value that the sum value of bounding variable int each cluster should be greater than.
`iterations`	(optional) The number of iterations of greedy algorithm. Defaults to 99.

`initial_regions`
: (optional) The initial regions that the local search starts with. Default is empty. means the local search starts with a random process to "grow" clusters.

`scale_method`
: (optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).

`distance_method`
: (optional) The distance method used to compute the distance betwen observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan"

`seed`
: (optional) The seed for random number generator. Defaults to 123456789.

`cpu_threads`
: (optional) The number of cpu threads used for parallel computation.Default is 6.

`rdist`
: (optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_maxp_greedy(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns',
'Infants','Suicids'),boundvar = 'Pop1831',min_bound = 3236.67)
guerry_clusters
```

`st_maxp_sa` *A simulated annealing algorithm to solve the max-p-region problem*

Description

A wrapper function for `rgeoda::maxp_sa()`.The max-p-region problem is a special case of constrained clustering where a finite number of geographical areas are aggregated into the maximum number of regions (max-p-regions), such that each region is geographically connected and the clusters could maximize internal homogeneity.

Usage

```
st_maxp_sa(
  sfj,
  varcol,
  wt = NULL,
  boundvar,
  min_bound,
  cooling_rate = 0.85,
  sa_maxit = 1,
  iterations = 99,
  initial_regions = vector("numeric"),
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  cpu_threads = 6,
  rdist = numeric()
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`boundvar`	A numeric vector of selected bounding variable.
`min_bound`	A minimum value that the sum value of bounding variable int each cluster should be greater than.
`cooling_rate`	(optional) The cooling rate of a simulated annealing algorithm. Defaults to 0.85.
`sa_maxit`	(optional) The number of iterations of simulated annealing. Defaults to 1.
`iterations`	(optional) The number of iterations of greedy algorithm. Defaults to 99.
`initial_regions`	(optional) The initial regions that the local search starts with. Default is empty. means the local search starts with a random process to "grow" clusters.
`scale_method`	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
`distance_method`	(optional) The distance method used to compute the distance betwen observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan"
`seed`	(optional) The seed for random number generator. Defaults to 123456789.
`cpu_threads`	(optional) The number of cpu threads used for parallel computation.Default is 6.
`rdist`	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_maxp_sa(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns',
'Infants','Suicids'),boundvar = 'Pop1831',min_bound = 3236.67)
guerry_clusters
```

---

st_maxp_tabu *A tabu-search algorithm to solve the max-p-region problem*

Description

A wrapper function for `rgeoda::maxp_tabu()`.The max-p-region problem is a special case of constrained clustering where a finite number of geographical areas are aggregated into the maximum number of regions (max-p-regions), such that each region is geographically connected and the clusters could maximize internal homogeneity.

Usage

```
st_maxp_tabu(
  sfj,
  varcol,
  wt = NULL,
  boundvar,
  min_bound,
  tabu_length = 10,
  conv_tabu = 10,
  iterations = 99,
  initial_regions = vector("numeric"),
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  cpu_threads = 6,
  rdist = numeric()
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
boundvar	A numeric vector of selected bounding variable.
min_bound	A minimum value that the sum value of bounding variable int each cluster should be greater than.
tabu_length	(optional) The length of a tabu search heuristic of tabu algorithm. Defaults to 10.
conv_tabu	(optional): The number of non-improving moves. Defaults to 10.
iterations	(optional) The number of iterations of greedy algorithm. Defaults to 99.
initial_regions	(optional) The initial regions that the local search starts with. Default is empty. means the local search starts with a random process to "grow" clusters.
scale_method	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
distance_method	(optional) The distance method used to compute the distance betwen observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan"
seed	(optional) The seed for random number generator. Defaults to 123456789.
cpu_threads	(optional) The number of cpu threads used for parallel computation.Default is 6.
rdist	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_maxp_tabu(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns',
'Infants','Suicids'),boundvar = 'Pop1831',min_bound = 3236.67)
guerry_clusters
```

st_redcap *Regionalization with dynamically constrained agglomerative clustering and partitioning(REDCAP)*

Description

A wrapper function for `rgeoda::redcap()`.REDCAP (Regionalization with dynamically constrained agglomerative clustering and partitioning) is developed by D. Guo (2008). Like SKATER, REDCAP starts from building a spanning tree with 4 different ways (single-linkage, average-linkage, ward-linkage and the complete-linkage). The single-linkage way leads to build a minimum spanning tree. Then,REDCAP provides 2 different ways (first-order and full-order constraining) to prune the tree to find clusters. The first-order approach with a minimum spanning tree is exactly the same with SKATER. In GeoDa and pygeoda, the following methods are provided: * First-order and Single-linkage * Full-order and Complete-linkage * Full-order and Average-linkage * Full-order and Single-linkage * Full-order and Ward-linkage.

Usage

```
st_redcap(
  sfj,
  varcol,
  k,
  wt = NULL,
  boundvar = NULL,
  method = "fullorder-averagelinkage",
  min_bound = 0,
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  cpu_threads = 6,
  rdist = numeric()
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`k`	The number of clusters.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`boundvar`	(optional) A data frame / tibble with selected bound variable.
`method`	(optional) "firstorder-singlelinkage", "fullorder-completelinkage", "fullorder-averagelinkage"(default),"fu singlelinkage", "fullorder-wardlinkage"
`min_bound`	(optional) A minimum bound value that applies to all clusters.

scale_method	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
distance_method	(optional) The distance method used to compute the distance between observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan".
seed	(int,optional) The seed for random number generator. Defaults to 123456789.
cpu_threads	(optional) The number of cpu threads used for parallel computation.
rdist	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_redcap(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns','Infants','Suicids'),
4,method = "fullorder-completelinkage")
guerry_clusters
```

---

st_schc *Spatially Constrained Hierarchical Clucstering (SCHC)*

Description

A wrapper function for `rgeoda::schc()`.Spatially constrained hierarchical clustering is a special form of constrained clustering, where the constraint is based on contiguity (common borders). The method builds up the clusters using agglomerative hierarchical clustering methods: single linkage, complete linkage, average linkage and Ward's method (a special form of centroid linkage). Meanwhile, it also maintains the spatial contiguity when merging two clusters.

Usage

```
st_schc(
  sfj,
  varcol,
  k,
  wt = NULL,
  boundvar = NULL,
  method = "average",
```

```
  min_bound = 0,
  scale_method = "standardize",
  distance_method = "euclidean",
  rdist = numeric()
)
```

Arguments

sfj	An sf (simple feature) object.
varcol	The variable selected to calculate spatial lag, which is a character.
k	The number of clusters.
wt	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
boundvar	(optional) A data frame / tibble with selected bound variable.
method	(optional) "single", "complete", "average"(default),"ward".
min_bound	(optional) A minimum bound value that applies to all clusters.
scale_method	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
distance_method	(optional) The distance method used to compute the distance between observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan".
rdist	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_schc(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns','Infants','Suicids'),
4,method = "complete")
guerry_clusters
```

st_skater *Spatial C(K)luster Analysis by Tree Edge Removal(SKATER)*

Description

A wrapper function for `rgeoda::skater()`.SKATER forms clusters by spatially partitioning data that has similar values for features of interest.

Usage

```
st_skater(
  sfj,
  varcol,
  k,
  wt = NULL,
  boundvar = NULL,
  min_bound = 0,
  scale_method = "standardize",
  distance_method = "euclidean",
  seed = 123456789,
  cpu_threads = 6,
  rdist = numeric()
)
```

Arguments

`sfj`	An sf (simple feature) object.
`varcol`	The variable selected to calculate spatial lag, which is a character.
`k`	The number of clusters.
`wt`	(optional) The spatial weights object,which can use `st_weights()` to construct,default is constructed by `st_weights(sfj,'contiguity')`.
`boundvar`	(optional) A data frame / tibble with selected bound variable.
`min_bound`	(optional) A minimum bound value that applies to all clusters.
`scale_method`	(optional) One of the scaling methods 'raw', 'standardize', 'demean', 'mad', 'range_standardize', 'range_adjust' to apply on input data. Default is 'standardize' (Z-score normalization).
`distance_method`	(optional) The distance method used to compute the distance between observation i and j. Defaults to "euclidean". Options are "euclidean" and "manhattan".
`seed`	(int,optional) The seed for random number generator. Defaults to 123456789.
`cpu_threads`	(optional) The number of cpu threads used for parallel computation.
`rdist`	(optional) The distance matrix (lower triangular matrix, column wise storage).

Value

A names list with names "Clusters", "Total sum of squares", "Within-cluster sum of squares", "Total within-cluster sum of squares", and "The ratio of between to total sum of squares".

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
library(sf)
guerry = read_sf(system.file("extdata", "Guerry.shp", package = "rgeoda"))
guerry_clusters = st_skater(guerry,c('Crm_prs','Crm_prp','Litercy','Donatns','Infants','Suicids'),4)
guerry_clusters
```

---

st_summary *Summary of Spatial Weights*

Description

Warpping the `summary()` function for spatial weights

Usage

```
st_summary(wt, ...)
```

Arguments

`wt`	A Weight object
`...`	summary optional parameters

Value

A summary description of an instance of Weight-class

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
## Not run:
library(sf)
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = read_sf(guerry_path)
queen_w = tidyrgeoda::st_weights(guerry,'contiguity')
st_summary(queen_w)

## End(Not run)
```

st_weights *Construct Spatial Weights*

Description

Create a spatial weights

Usage

```
st_weights(sfj, weight = NULL, ...)
```

Arguments

sfj	An sf (simple feature) object.
weight	The method used to create spatial weights,which has to be one of 'contiguity', 'distance', 'knn', 'kernel', 'kernel_knn'.
...	Other arguments to construct spatial weight, see 'tidyrgeoda::st_contiguity_weights','tidyrgeoda::st_distan 'tidyrgeoda::st_knn_weights','tidyrgeoda::st_kernel_weights', 'tidyrgeoda::st_kernel_knn_weights'.

Value

An instance of rgeoda Weight-class.

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Examples

```
guerry_path = system.file("extdata", "Guerry.shp", package = "rgeoda")
guerry = sf::read_sf(guerry_path)
st_weights(guerry,'kernel_knn',6)
```

write_geoda *Save Spatial Weights*

Description

Save spatial weights to a file

Usage

```
write_geoda(wt, dsn, id_vec = NULL, layer = NULL)
```

Arguments

`wt`	A Weight object
`dsn`	The path of an output weights file
`id_vec`	(optional) Defines the unique value of each observation when saving a weights file. Default is `tibble::tibble(id_v = 1:wt$num_obs)`.
`layer`	(optional) The name of the layer of input dataset,default is `"tbf"`.

Value

A boolean value indicates if save successfully or failed

Author(s)

Wenbo Lv <lyu.geosocial@gmail.com>

Index