Package ‘mobicountR’

October 30, 2019

Type Package

Title Improving present population counts using mobile phone data

Version 0.1.0

Description
a flexible bayesian framework to improve present population counts using mobile phone data.

License CC0

Encoding UTF-8

LazyData true

Imports minior,
dplyr,
rlang,
foreach,
sf,
readr,
stringr,
tidyr,
data.table,
leaflet,
RColorBrewer

RoxygenNote 6.1.1

R topics documented:

bay_coverage *Computes the bayesian posterior distribution using a coverage map*

Description

This function implements the bayesian model for mobile events detection, assuming that a coverage map

Usage

```
bay_coverage(df_inter, prior)
```

Arguments

priors	a list of data.frames, one for each prior. See 'Details' for required structure
weights	numeric vector of same length as priors; weights used to compute the linear combination of priors. Must sum to one for the output to be a probability distribution.

Details

data.frames in priors should have the same structure as those produced by prior_building and prior_res_pop :
a first column indicating tiles IDs, and a second column giving the distribution of the prior on the grid.

Value

a data.frame with two columns : tiles ID and a prior distribution over tiles.

Examples

```
## Not run:
# Import priors
prior_bdtopo <- prior_building(paths_shp = "~/bdtopo_shp", grid = grid_500_france)
prior_rfl <- prior_res_pop(rfl = rfl11)
# Include inputs in a list
list_priors <- list(prior_bdtopo, prior_rfl)
# Define weights to be used (same order as the list of priors)
weights = c(0.3, 0.7)
# Combine priors
prior_comb <- prior_combine(priors = list_priors, weights = weights)

## End(Not run)
```

bay_voronoi — *Computes the bayesian posterior distribution using a voronoi tesselation*

Description

This function implements the bayesian model for mobile events detection, assuming that no information on the MNO's antennas coverage is available. A voronoi tesselation is thus used to approximate the area covered by each antenna. If available, prior information over tiles can be used to improve the quality of the detection.

Usage

```
bay_voronoi(df_inter, prior = NULL, var_prior = NULL,
  var_final = "proba_final”, filter_null = TRUE)
```

Arguments

`df_inter` a `data.frame` which indicates for each voronoi x grid intersection the relative area of the intersection in the corresponding voronoi; should be either produced by or have the same format as the output of `inter_grid_voronoi`.

`prior` a `data.frame`; should be either produced by or have the same format as those produced by the prior computation functions of `mobicount`. Defauts to `NULL`, in which case a prior equal to one is used (uniform).

`var_prior` character; name of the variable which contains the prior distribution; ignored if `prior = NULL`.

`var_final` character; name of the variable which should contain the posterior distribution.

`filter_null` logical; should voronoi x grid intersections with a null posterior probability be deleted ? Defauts to `TRUE`.

Value

a `data.frame` with two columns : an ID for each voronoi x grid intersection formatted as "voronoi_ID:grid_id", and the posterior distribution.

Examples

```
## Not run:
# Import prior
prior_bdtopo <- readr::read_csv(~/bdtopo_france.csv)
# Import voronoi x grid intersections table
inter_df <- readr::read_csv(~/inter_grid_voro.csv)
# Compute bayesian posterior distribution
bay_df_bdtopo <- bay_voronoi(inter_df, prior_bdtopo, ”proba_bdtopo”)

## End(Not run)
```

`create_grid` *Creates a regular grid over a spatial unit*

Description

Creates a regular square grid over the bounding box of an sf object (e.g. a country limits shapefile imported with `st_read`).

Usage

```
create_grid(x, tile_size, crs = 2154)
```

Arguments

`x` data.frame of class `sf`.

`tile_size` integer; length of the side of a square (tile).

`crs` integer; desired projected coordinate system; defauts to 2154 (Lambert 93).

Details

The output data.frame contains a variable grid_id which provides an unique identifier for each tile. Identifiers are constructed as x_centroid/100:y_centroid/100 so that tile centroids are easy to compute back when needed.

Value

the created grid as a data.frame of class `sf`.

Examples

```
## Not run:
# Import shapefile of country limits
fr_limits <- sf::st_read("~/france_shp/francemetro_2015.shp")
# Create grid
grid <- create_grid(x = fr_limits, tile_size = 500)

## End(Not run)
```

`inter_grid_voronoi` *Computes intersections between grid and voronoi tesselation*

Description

Computes intersections between a grid (e.g. created with `create_grid`) and a voronoi tesselation computed over an MNO antennas map. Intersections probabilities, defined as the relative area of an intersection in the correspond voronoi, are computed during the process.

Usage

```
inter_grid_voronoi(grid, voronoi, grid_id = "grid_id",
  voronoi_id = "NIDT", sf = FALSE, crs = 2154)
```

Arguments

`grid` data.frame of class `sf`.

`voronoi` data.frame of class `sf`.

`sf` logical; if `TRUE`, output is an sf object; defauts to `FALSE`.

`crs` integer; desired projected coordinate system; defauts to 2154 (Lambert 93).

Details

The output data.frame contains a variable `proba_inter` which provides, for each intersection between a tile and a voronoi, the relative area of the tile in the corresponding voronoi. This value is comprised between 0 and 1 and thus assimilated to a probability of intersection.

Value

a simple data.frame if `sf=TRUE`. A data.frame of class `sf` if `sf=FALSE`.

Examples

```
## Not run:
# Create grid
grid_sf <- create_grid(x = fr_limits, tile_size = 500, export = FALSE)
# Import shapefile of voronoi tesselation
voronoi_sf <- sf::st_read("~/Antenne_voronoi_rev.shp", crs = 2154)
# Compute grid-voronoi intersections
table_prob <- inter_grid_voronoi(grid = grid_sf,
voronoi = voronoi_sf, prob = TRUE, sf = FALSE)

## End(Not run)
```

`mobicount` *Improving present population counts using mobile phone data*

Description

mobicount provides a flexible bayesian framework to improve present population counts using mobile phone data.

Author(s)

Romain Avouac

`prior_building` *Computes prior from building registers*

Description

Computes a prior distribution over a grid using data from building registers (e.g. shapefiles of buildings over the territory).

Usage

```
prior_building(paths_shp, grid, grid_id_var = "grid_id",
  dir_inter = paste0(tempdir(), "/dir_inter"), area_min_max = NULL,
  height_var = "HAUTEUR", impute = TRUE, height_min_max = c(NA, 130),
  parallel = TRUE, n_workers = 5, crs = 2154)
```

Arguments

`paths_shp` character vector; complete paths to building shapefiles.

`grid` a `data.frame` of class `sf`.

`grid_id_var` character; name of the variable indicating tiles unique identifier in `grid`.

`dir_inter` character; path to the desired output directory for prior files.

`area_min_max` numeric vector of length 2; minimum and/or maximum areas allowed for buildings, outside of which buildings are filtered. To provide only one value, indicate the other as `NA`.

`height_var` character; name of the variable indicating building heights. If set to `NULL`, prior is computed using only building areas.

`impute` logical; should heights of zero height buildings be imputed ?

`height_min_max` numeric vector of length 2; minimum and/or maximum heights allowed for buildings, outside of which buildings are filtered. To provide only one value, indicate the other as `NA`. Minimum height is ignored if `impute = TRUE`.

`parallel` logical; if multiple building shapefiles are provided, should the computations on each of these files be run in parallel ?

`n_workers` number of worker processes that `doParallel` will use to execute tasks in parallel. Ignored if `parallel` = `FALSE`.

`crs` integer; desired projected coordinate system; defauts to 2154 (Lambert 93).

Details

If multiple shapefiles are provided (e.g. shapefiles at an infra-national level), prior computation can be computed in parallel using `foreach`.

If the provided grid was not computed using `create_grid`, tiles ID should be formatted the same way : "x_centroid/100:y_centroid/100".

Value

A `data.frame` of class `data.table` with two columns : tiles ID and a prior distribution over tiles.

Examples

```
## Not run:
# Import building shapefiles paths
bdtopo_shp <- list.files("~/bdtopo")
# Compute grid
grid_500_fr <- create_grid(x = fr_limits, tile_size = 500)
# Compute prior
build_prior_dt <- prior_building(paths_shp = bdtopo_shp, grid = grid_500_fr)

## End(Not run)
```

`prior_combine` *Combines multiple priors*

Description

Combines multiple prior distributions over a same grid by computing a new distribution as a linear combination

Usage

```
prior_combine(priors, weights)
```

Arguments

`priors` a list of `data.frames`, one for each prior. See 'Details' for required structure

`weights` numeric vector of same length as `priors`; weights used to compute the linear combination of priors.
Must sum to one for the output to be a probability distribution.

Details

`data.frames` in `priors` should have the same structure as those produced by prior_building and prior_res_pop :
a first column indicating tiles IDs, and a second column giving the distribution of the prior on the grid.

Value

a `data.frame` with two columns : tiles ID and a prior distribution over tiles.

Examples

```
## Not run:
# Import priors
prior_bdtopo <- prior_building(paths_shp = "~/bdtopo_shp", grid = grid_500_france)
prior_rfl <- prior_res_pop(rfl = rfl11)
# Include inputs in a list
list_priors <- list(prior_bdtopo, prior_rfl)
# Define weights to be used (same order as the list of priors)
weights = c(0.3, 0.7)
# Combine priors
prior_comb <- prior_combine(priors = list_priors, weights = weights)

## End(Not run)
```

`prior_res_pop` *Computes prior from resident population*

Description

Computes a prior distribution over a grid using resident population data from fiscal localized income sources (e.g. RFL or FiLoSoFi data in France).

Usage

```
prior_res_pop(rfl, indiv_id_var = "DIRNOSEQ", n_indiv_var = "nbpersm",
  x_y_var = c("x", "y"))
```

Arguments

`rfl` a `data.frame` of class `data.table` containing fiscal localized incomes data. If a simple `data.frame` is provided, it is automatically converted to a `data.table`.

`indiv_id_var` character; name of the variable indicating household unique identifier in `rfl`.

`n_indiv_var` character; name of the variable indicating household sizes (number of individuals) in `rfl`.

`x_y_var` character vector of length 2; names of the variables indicating x and y coordinates of the household in `rfl`.

Details

If the output data.frame of this function is to be used as in , x and y coordinates in input data (rfl) should be converted beforehand in Lambert-93 (ESPG:2154).

In the output data.frame, tiles ID are formatted as "x_centroid/100:y_centroid/100" in order to match the formatting of grids created by create_grid.

Value

a data.frame of class data.table with two columns : tiles ID and a prior distribution over tiles.

Examples

```
## Not run:
library(sas7bdat)
# Import data from fiscal localized data in 2014
rfl_df <- read.sas7bdat("path_to_filosofi_2014/menages14.sas7bdat")
# Compute prior using resident population
res_prop_prior <- prior_res_pop(rfl = rfl_df, indiv_id_var = "IDENTIFIANT")

## End(Not run)
```

---

sfc_as_cols *Transforms a sf POINT geometry to x,y columns*

Description

Transforms a data.frame of class sf with a POINT geometry (e.g. a grid of which centroids has been computed using st_centroid) to a conventional data.frame with coordinates as two numeric columns.

Usage

```
sfc_as_cols(x, names = c("x", "y"))
```

Arguments

x	data.frame of class sf with a sfc_POINT geometry.
names	character vector of length 2; names of the two new columns with x,y coordinates; defauts to c("x","y").

Details

This transformation is useful for efficient filtering of coordinates, e.g. to subset a grid to a smaller area bounding box.

Source of the function : https://github.com/r-spatial/sf/issues/231

Value

A data.frame with two new columns corresponding to x,y coordinates of spatial units. The initial geometry column is removed.

Examples

```
## Not run:
# Import grid and compute centroids
grid <- sf::st_read("~/grid_500_france.shp")
grid <- grid %>% st_centroid()
# Transform to a data.frame with centroid coordinates as new columns
grid_nogeo <- grid %>% sfc_as_cols()

## End(Not run)
```

Index