Package: nyped (via r-universe)

June 5, 2026

Title Model of pedestrian flows throughout New York City

Version 0.0.0.017

Description Model of pedestrian flows throughout New York City.

License GPL-3

Encoding UTF-8

LazyData true

Imports cli, dodgr, dplyr, geodist, magrittr, methods, moveability, osmdata, raster, Rcpp (>= 0.12.6), rvest, sf, xml2

Suggests mapdeck, mapview, testthat

LinkingTo Rcpp

NeedsCompilation yes

URL https://github.com/ATFutures/nyped

BugReports https://github.com/ATFutures/nyped/issues

RoxygenNote 7.0.0

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libicu-dev libuv1-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2020-01-16 20:49:57 UTC

RemoteUrl https://github.com/ATFutures/nyped

RemoteRef HEAD

RemoteSha 6b0e3a64452d76654b693d99016a5616ae690288

Contents

`add_layer_to_model` *add_layer_to_model*

Description

Build final pedestrian model

Usage

```
add_layer_to_model(data_dir, dat = NULL, sig = 0.01, pos_only = TRUE)
```

Arguments

`data_dir`	The directory in which data are to be, or have previously been, downloaded.
`dat`	Output of previous run of 'add_layer_to_model'
`sig`	Desired level of statistical significance
`pos_only`	Only include layers that make a positive contribution to final model (and so exclude any layers that are negatively correlated)?

all_flows_to_ped *all_flows_to_ped*

Description

Batch-convert all flow layers to equivalent values matched to pedestrian counting stations via the fit_flows_to_ped function.

Usage

```
all_flows_to_ped(data_dir)
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.

build_ped_model *build_ped_model*

Description

build_ped_model

Usage

```
build_ped_model(data_dir, sig_build = 0.05, sig_model = 0.01, pos_only = FALSE)
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.
sig_build	Significance threshold for initial constructing of model through adding successive layers
sig_model	Desired significance of final model
pos_only	Only include layers that make a positive contribution to final model (and so exclude any layers that are negatively correlated)?

Value

The final model reduced to only layers with significance below ‘sig_model‘.

Note

This function can take several minutes to run, and saves the final model in two forms, both stored in ‘data_dir‘: the initial, full model with all layers, called ’ped-model-full.Rds’, and the final model with only layers with significance below ‘sig_model‘, called ‘ped-model-final.Rds‘.

`calc_layer_scales` *calc_layer_scales*

Description

Calculate relative scales of contributions of each flow layer to final model of pedestrian flows generated by successive calls to build_ped_model.

Usage

```
calc_layer_scales(data_dir, model = "final")
```

Arguments

`data_dir`	The directory in which data are to be, or have previously been, downloaded.
`model`	Default "final" uses the final model of layers reduced only to those with significance less than ‘sig_model‘ in build_ped_model, otherwise the full model with potentially less significant layers is used.

Value

A ‘data.frame‘ with six columns of (1) ’model’ with the layer names; (2) ’estimates’ holding the estimates for each layer from the multiple linear regression model; (3) ’full_rel’ with the relative scaling coefficient for the contribution of each layer weighted by the full flow layers; (4) ’ped_rel’ with equivalent values weighted by values at pedestrian count stations only; (5) ’full_abs’ with the absolute scaling coefficient for each layer weighted by full flow layers; and (6) ’ped_abs’ with equivalent values weighted by values at pedestrian count stations only.

`centrality_edge_to_point` *centrality_edge_to_point*

Description

Convert edge-based centrality measures to equivalent point-based.

Usage

```
centrality_edge_to_point(data_dir, save = FALSE)
```

Arguments

`data_dir`	The data directory
`save`	If ‘TRUE‘, save resultant vertex-based centrality measures in ‘data_dir‘.

Value

Equivalent point- or vertex-based measures of centrality

`clb_osm_data` *clb_osm_data*

Description

Get OpenStreetMap calibration data.

Usage

```
clb_osm_data(city, prefix, data_dir = NULL)
```

Arguments

`city`	City or bounding box for which data are to be extracted
`prefix`	Prefix to prepend to file names to identify city
`data_dir`	Directory in which data are to be stored

Value

List of OSM data for highways, green spaces, and activity attractors.

`cut_network_to_pts` *cut_network_to_pts*

Description

Cut the New York City street network into portions surrounding each pedestrian counting point.

Usage

```
cut_network_to_pts(net, k = 1000, p = NULL, data_dir)
```

Arguments

`net`	Weighted street network; loaded from ‘data_dir‘ if not provided
`k`	Exponential decay parameter to be used in spatial interaction models. This deterines the radius at which to cut the network.
`p`	Pedestrian counts including OSM IDs of nearest points, as returned from ped_osm_id.
`data_dir`	The directory in which data are to be, or have previously been, downloaded - only needed if ‘p‘ is not provided.

`fit_cen_to_ped` *fit_cen_to_ped*

Description

Fit network centrality to pedestrian counts, by aggregating across a variable number of edges ('n') nearest to each pedestrian count station.

Usage

```
fit_cen_to_ped(data_dir)
```

Arguments

`data_dir`	The directory in which data are to be, or have previously been, downloaded.

Value

A list containing vectors of 'k' and 'n' values, and a matrix of 30x20 = 600 columns, one for each combination of 30 'k'- and 20 'n'-values.

`fit_flows_to_ped` *fit_flows_to_ped*

Description

Fit one network with multiple flow columns (from multiple k-values) to pedestrian counts, by aggregating each flow column across a variable number of edges ('n') nearest to each pedestrian count station.

Usage

```
fit_flows_to_ped(net_f, data_dir)
```

Arguments

`net_f`	A network with flow columns, obtained from 'get_layer'
`data_dir`	The directory in which data are to be, or have previously been, downloaded.

Value

A list containing vectors of 'k' and 'n' values, and a matrix of 30x20 = 600 columns, one for each combination of 30 'k'- and 20 'n'-values.

get_attractors *get_attractors*

Description

Get points of trip attraction

Usage

```
get_attractors(bbox, quiet = FALSE)
```

Arguments

bbox	Bounding box for which green space polygons are to be extracted.
quiet	If 'TRUE', dump progress information to screen.

get_green_space *get_green_space*

Description

Get polygons of all green areas for a given location

Usage

```
get_green_space(bbox, quiet = FALSE)
```

Arguments

bbox	Bounding box for which green space polygons are to be extracted.
quiet	If 'TRUE', dump progress information to screen.

Value

An **sf**-format 'data.frame' of polygons representing all green areas.

get_layer *get_layer*

Description

Append network with flow columns between nominated places

Usage

```
get_layer(net, from = "subway", to = "disperse", data_dir)
```

Arguments

net	Weighted street network; loaded from ‘data_dir‘ if not provided
from	Category of origins for pedestrian flows; one of "subway" or "residential"
to	Category of destinations for pedestrian flows; one of "residential", "education", "entertainment", "healthcare", "sustenance", "transportation", or "disperse" for a general dispersal model.
data_dir	The directory in which data are to be, or have previously been, downloaded.

nyosm_data *nyosm_data*

Description

Get OpenStreetMap data for New York City

Usage

```
nyosm_data(data_dir)
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.

Value

A list of three items: (1) the street network; (2) building polygons; and (3) green space polygons

nyped_data *nyped_data*

Description

Download, clean, and return New York City pedestrian count data

Usage

```
nyped_data(data_dir = tempdir(), quiet = FALSE)
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.
quiet	If 'FALSE', display progress information on screen

Value

A 'data.frame' of pedestrian counts, and geographical coordinates, with counts for weekdays, weekends, and "week" derived as a weighted combintaion of both.

Examples

```
dat <- nyped_data ()
# library (mapview)
# mapview (dat, cex = "week", zcol = "week")
```

nypopdens *nypopdens*

Description

Get population density data for New York City

Usage

```
nypopdens(data_dir = tempdir())
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.

Value

Population density data

nysubway_data *nysubway_data*

Description

Download, clean, and join New York City subway station counts and coordinates

Usage

```
nysubway_data(quiet = FALSE, sub_exits = TRUE)
```

Arguments

quiet	If 'FALSE', display progress information on screen
sub_exits	Calculate layer from subway exits ('TRUE'), or just from single points denoting subway stations ('FALSE')?

Value

A 'data.frame' of subway names, annual counts, and geographical coordinates.

Examples

```
dat <- nysubway_data ()
# library (mapview)
# mapview (dat, cex = "count2018", zcol = "count2018")
```

ped_model_to_full_flow
ped_model_to_full_flow

Description

Convert a final model of pedestrian flows at the pedestrian count stations back into a model of full flows along each edge of the entire network

Usage

```
ped_model_to_full_flow(mod, data_dir)
```

Arguments

mod	A final pedestrian model produced by successive calls to build_ped_model
data_dir	The directory in which data are to be, or have previously been, downloaded.

ped_osm_id *ped_osm_id*

Description

Get OSM IDs nearest to pedestrian count points

Usage

```
ped_osm_id(data_dir, net = NULL, quiet = FALSE)
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.
net	Weighted street network; loaded from ‘data_dir‘ if not provided
quiet	If ‘FALSE‘, display progress information on screen

Value

A ‘data.frame‘ of pedestrian counts, associated spatial coordinates, and OSM IDs of nearest points on network

plot_ny_popdens *plot_ny_popdens*

Description

Plot the population density of New York City via ’mapdeck’

Usage

```
plot_ny_popdens(data_dir)
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.

Note

This function presumes that a mapdeck token exists as an environmental variable with a name that includes "mapbox"

`plot_ped_counts` *plot_ped_counts*

Description

Plot pedestrian count data for New York City

Usage

```
plot_ped_counts(type = "week")
```

Arguments

`type`	One of "weekday", "weekend", or "week" (aggregate of both)

`plot_subway_counts` *plot_subway_counts*

Description

Plot subway entrace/exit count data for New York City

Usage

```
plot_subway_counts(year = 2018L)
```

Arguments

`year`	Year in [2013:2018]

`subway_osm_id` *subway_osm_id*

Description

Get OSM IDs nearest to subway count points

Usage

```
subway_osm_id(data_dir, net = NULL, sub_exits = TRUE, quiet = FALSE)
```

Arguments

data_dir	The directory in which data are to be, or have previously been, downloaded.
net	Weighted street network; loaded from 'data_dir' if not provided
sub_exits	Calculate layer from subway exits ('TRUE'), or just from single points denoting subway stations ('FALSE')?
quiet	If 'FALSE', display progress information on screen

Value

A 'data.frame' of pedestrian counts, associated spatial coordinates, and OSM IDs of nearest points on network

Index