Package: ttcalib (via r-universe)

November 13, 2024

Title Calibration of travel times to empirical data

Version 0.1.0.009

Description Calibration of travel times to empirical data.

License GPL-3

URL https://github.com/UrbanAnalyst/ttcalib

BugReports https://github.com/UrbanAnalyst/ttcalib/issues

Depends R (>= 2.10)

Imports cli, dodgr, dplyr, fst, m4ra, readr, sf

Suggests geodist, testthat (>= 3.0.0)

Remotes UrbanAnalyst/m4ra

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.2

Config/testthat/edition 3

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev make libicu-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libx11-dev

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

RemoteUrl https://github.com/UrbanAnalyst/ttcalib

RemoteRef HEAD

RemoteSha 1858d0d1ba954394172eb54eb79cb6a8000d1079

Contents

`ttcalib_centrality` *Calibrate travel times to network centrality.*

Description

This calibration step is performed after calibration to waiting-time penalties for traffic lights and turning across oncoming traffic, performed with the ttcalib_penalties function. For Santiago, that function gives an optimal waiting time at traffic lights of 16 seconds, and waiting time to turn across oncoming traffic of 1 second. These values should be used to generate a weighted network with time-based centrality via the ttcalib_streetnet function.

Usage

```
ttcalib_centrality(
  path_graph,
  path_uberdata,
  city = "santiago",
  hours = c(7, 10),
  turn_penalty = 1
)
```

Arguments

`path_graph`	Path to locally-saved `fst`-format weighted street network including centrality column.
`path_uberdata`	Path to Uber movement data.
`city`	Currently only accepts "santiago"
`hours`	A vector of two values defining the range of hours for Uber Movement data to be filtered. Value of `NULL` aggregates all hours without filtering.
`turn_penalty`	The value of the time penalty for waiting to turn across oncoming traffic used to generate the graph stored at `path`.

Details

The result of that call is presumed to have been saved using the `fst` package (with `write_fst`), which strips all attributes of the graph. These attributes must then be manually re-instated, and so are required to be submitted as parameters to this function.

ttcalib_geodata *Read geometries for chosen city from Uber Movement data.*

Description

Currently hard-coded to Brussels or Santiago data only.

Usage

```
ttcalib_geodata(path, city = "santiago")
```

Arguments

path	Path to directory containing Uber movement data.
city	One of "brussels" or "santiago" (case-insensitive).

Value

A 'data.frame' with centroids of polygons used to aggregate movement data.

ttcalib_penalties *Calculate standard error against 'uber' data for all weighted networks generated via ttcalib_streetnet_batch.*

Description

Calculate standard error against 'uber' data for all weighted networks generated via ttcalib_streetnet_batch.

Usage

```
ttcalib_penalties(path_results, path_uberdata, city, hours = NULL)
```

Arguments

path_results	Path to directory holding main OSM network data, including a sub-directory with output of ttcalib_streetnet_batch.
path_uberdata	Path to Uber movement data.
city	One of "brussels" or "santiago" (case-insensitive).
hours	A vector of two values defining the range of hours for data to be filtered. Default of NULL returns aggregate of all hours without filtering.

`ttcalib_streetnet` *Load an 'SC' street network, weight for motorcar routing, and optionally calculate centrality.*

Description

Load an 'SC' street network, weight for motorcar routing, and optionally calculate centrality.

Usage

```
ttcalib_streetnet(
  path,
  centrality = FALSE,
  penalty_traffic_lights = 8,
  penalty_turn = 7.5,
  dist_threshold = 10000
)
```

Arguments

`path`	Path to 'SC'-format file containing street network data.
`centrality`	If `TRUE`, calculate network centrality on all graph edges. Load an 'SC' street network, weight for motorcar routing, and calculate centrality.
`penalty_traffic_lights`	Time penalty for waiting at traffic lights (in seconds).
`penalty_turn`	Time penalty for turning across oncoming traffic.
`dist_threshold`	Threshold used for centrality calculations (in metres); see documentation for **dodgr** function, 'dodgr_centrality' for information.

Value

Network with centrality estimates on each edge.

`ttcalib_streetnet_batch` *Weight an 'SC' street network by a range of traffic light penalties, and locally save the results.*

Description

This function produces an array of differently-weighted streetnets, each locally saved to a uniquely-named file. The resultant graphs are saved with the **fst** package, which removes the necessary attributes of the resultant `data.frame` objects. These must be manually restored prior to submitting to ttcalib_traveltimes. Required attributes are:

- "left_side", passed as same parameter to **dodgr** function `wt_streetnet`.
- "wt_profile", passed as same parameter to **dodgr** function `wt_streetnet`, and which should generally be "motorcar" here.

Usage

```
ttcalib_streetnet_batch(path, penalty_traffic_lights = 1:10)
```

Arguments

`path`	Path to 'SC'-format file containing street network data.
`penalty_traffic_lights`	Time penalty for waiting at traffic lights (in seconds).

Note

This function only generated weighted networks for a range of traffic light penalties. Turn penalties are calculated on-the-fly in **m4ra** for each routing call, and can be specified by modifying the "turn_penalty" attribute of the graph.

---

ttcalib_traveltimes *Estimate travel times between all Uber movement polygons*

Description

Estimate travel times between all Uber movement polygons

Usage

```
ttcalib_traveltimes(graph, geodata, uberdata)
```

Arguments

`graph`	A **dodgr** graph returned from ttcalib_streetnet.
`geodata`	A `data.frame` returned from ttcalib_geodata.
`uberdata`	A `data.frame` returned from ttcalib_uberdata.

Value

A 'data.frame' with columns of **m4ra** estimates of travel times and corresponding empirical values from Uber movement data.

`ttcalib_uberdata` *Read Uber Movement data and filter to defined time limits.*

Description

Currently hard-coded to Brussels data only.

Usage

```
ttcalib_uberdata(path, city = "santiago", hours = NULL)
```

Arguments

`path`	Path to directory containing Uber movement data.
`city`	One of "brussels" or "santiago" (case-insensitive).
`hours`	A vector of two values defining the range of hours for data to be filtered. Default of `NULL` returns aggregate of all hours without filtering.

Value

A 'data.frame' of Uber Movement estimates of travel times.

Index