Package ‘cyclestreets’

October 12, 2022

Type Package
Title Cycle Routing and Data for Cycling Advocacy
Version 0.5.3
Description An interface to the cycle routing/data services provided by 'CycleStreets', a not-for-profit social enterprise and advocacy organisation. The application programming interfaces (APIs) provided by 'CycleStreets' are documented at (<https://www.cyclestreets.net/api/>). The focus of this package is the journey planning API, which aims to emulate the routes taken by a knowledgeable cyclist. An innovative feature of the routing service of its provision of fastest, quietest and balanced profiles. These represent routes taken to minimise time, avoid traffic and compromise between the two, respectively.

License GPL-3
BugReports https://github.com/cyclestreets/cyclestreets-r/issues
Depends R (>= 3.6.0)
Imports geodist, httr, jsonlite, magrittr, sf, stringr
Suggests covr, stplanr
Encoding UTF-8
LazyData true
RoxygenNote 7.1.2
NeedsCompilation no
Author Robin Lovelace [aut, cre] (<https://orcid.org/0000-0001-5679-6536>), Martin Lucas-Smith [aut], Eric Krueger [ctb], Joey Talbot [aut] (<https://orcid.org/0000-0002-6520-4560>)
Maintainer Robin Lovelace <rob00x@gmail.com>
Repository CRAN
Date/Publication 2022-01-04 23:10:02 UTC
## R topics documented:

<table>
<thead>
<tr>
<th>Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cyclestreets_column_names</td>
<td>2</td>
</tr>
<tr>
<td>journey</td>
<td>2</td>
</tr>
<tr>
<td>json2sf_cs</td>
<td>5</td>
</tr>
<tr>
<td>ltns</td>
<td>6</td>
</tr>
<tr>
<td>smooth_with_cutoffs</td>
<td>7</td>
</tr>
</tbody>
</table>

### cyclestreets_column_names

*Prices of 50,000 round cut diamonds.*

### Description

Variables provided by CycleStreets in their journey data

### Usage

```r
cyclestreets_column_names
```

### Format

An object of class character of length 44.

### Source

[https://www.cyclestreets.net/](https://www.cyclestreets.net/)

---

### journey

*Plan a journey with CycleStreets.net*

### Description

R interface to the CycleStreets.net journey planning API, a route planner made by cyclists for cyclists. See cyclestreets.net/api for details.
journey

Usage

journey(
  from,
  to,
  plan = "fastest",
  silent = TRUE,
  pat = NULL,
  base_url = "https://www.cyclestreets.net",
  reporterrors = TRUE,
  save_raw = "FALSE",
  cols = c("name", "distances", "time", "busynance", "elevations", "start_longitude",
           "start_latitude", "finish_longitude", "finish_latitude"),
  cols_extra = c("crow_fly_distance", "event", "whence", "speed", "itinerary",
                 "clientRouteId", "plan", "note", "length", "quietness", "west", "south", "east",
                 "north", "leaving", "arriving", "grammesCO2saved", "calories", "edition",
                 "gradient_segment", "elevation_change", "provisionName"),
  smooth_gradient = TRUE,
  distance_cutoff = 50,
  gradient_cutoff = 0.1,
  n = 3
)

Arguments

from    Longitude/Latitude pair, e.g. c(-1.55, 53.80)
to      Longitude/Latitude pair, e.g. c(-1.55, 53.80)
plan    Text strong of either "fastest" (default), "quietest" or "balanced"
silent  Logical (default is FALSE). TRUE hides request sent.
pat     The API key used. By default this uses Sys.getenv("CYCLESTREETS").
base_url The base url from which to construct API requests (with default set to main server)
reporterrors Boolean value (TRUE/FALSE) indicating if cyclestreets (TRUE by default). should report errors (FALSE by default).
save_raw Boolean value which returns raw list from the json if TRUE (FALSE by default).
cols    Columns to be included in the result, a character vector or NULL for all available columns (see details for default)
cols_extra Additional columns to be added providing summaries of gradient and other variables
smooth_gradient Identify and fix anomalous gradients? TRUE by default. See https://github.com/Robinlovelace/cyclestreets
distance_cutoff Distance (m) used to identify anomalous gradients
gradient_cutoff Gradient (%. e.g. 0.1 being 10%) used to identify anomalous gradients
n      The number of segments to use to smooth anomalous gradients. The default is 3, meaning segments directly before, after and including the offending segment.
Details

Requires the internet and a CycleStreets.net API key. CycleStreets.net does not yet work worldwide.

You need to have an api key for this code to run. By default it uses the CYCLESTREETS environment variable. A quick way to set this is to install the usethis package and then executing the following command:

```r
usethis::edit_r_environ()
```

That should open up a new file in your text editor where you can add the environment variable as follows (replace 1a... with your key for this to work):

```
CYCLESTREETS=1a43ed677e5e6fe9
```

After setting the environment variable, as outlined above, you need to restart your R session before the journey function will work.

A full list of variables (cols) available is represented by:

```
c("time", "busynance", "signalledJunctions", "signalledCrossings", 
"name", "walk", "elevations", "distances", "start", "finish", 
"startSpeed", "start_longitude", "start_latitude", "finish_longitude", 
"finish_latitude", "crow_fly_distance", "event", "whence", "speed", 
"itinerary", "clientRouteId", "plan", "note", "length", "quietness", 
"west", "south", "east", "north", "leaving", "arriving", "grammesCO2saved", 
"calories", "edition", "geometry")
```

See [www.cyclestreets.net/help/journey/howitworks/](http://www.cyclestreets.net/help/journey/howitworks/) for details on how these are calculated.

See Also

`json2sf_cs`

Examples

```r
## Not run:
from = c(-1.55, 53.80) # geo_code("leeds")
to = c(-1.76, 53.80) # geo_code("bradford uk")
r1 = journey(from, to)
names(r1)
r1[1:2, ]
r1$grammesCO2saved
r1$calories
plot(r1[1:4])
plot(r1[10:ncol(r1)])
to = c(-2, 53.5) # towards Manchester
r1 = journey(from, to)
names(r1)
r2 = journey(from, to, plan = "balanced")
plot(r1["quietness"], reset = FALSE)
plot(r2["quietness"], add = TRUE)
r3 = journey(from, to, silent = FALSE)
r4 = journey(from, to, save_raw = TRUE)
r5 = journey(c(-1.524, 53.819), c(-1.556, 53.806))
```
plot(r5[["gradient_segment"]])
plot(r5[["gradient_smooth"]])

u = paste0("https://github.com/cyclestreets/cyclestreets-r/",
  "releases/download/v0.4.0/line_with_single_segment.geojson")
desire_line = sf::read_sf(u)
r = stplanr::route(l = desire_line, route_fun = journey)
r

## End(Not run)

---

convert_from_cyclestreets

Convert output from CycleStreets.net into sf object

**Description**

Convert output from CycleStreets.net into sf object

**Usage**

```r
json2sf_cs(
  obj,
  cols = NULL,
  cols_extra = c("elevation_start", "elevation_end", "gradient_segment",
    "elevation_change", "provisionName"),
  smooth_gradient = FALSE,
  distance_cutoff = 50,
  gradient_cutoff = 0.1,
  n = 3
)
```

**Arguments**

- **obj**: Object from CycleStreets.net read-in with
- **cols**: Columns to be included in the result, a character vector or NULL for all available columns (see details for default)
- **cols_extra**: Additional columns to be added providing summaries of gradient and other variables
- **smooth_gradient**: Identify and fix anomalous gradients? TRUE by default. See https://github.com/Robinlovelace/cyclestreets
- **distance_cutoff**: Distance (m) used to identify anomalous gradients
- **gradient_cutoff**: Gradient (%) used to identify anomalous gradients
- **n**: The number of segments to use to smooth anomalous gradients. The default is 3, meaning segments directly before, after and including the offending segment.
Examples

```r
from = "Leeds Rail Station"
to = "University of Leeds"
# from_point = tmaptools::geocode_OSM(from)
# to_point = tmaptools::geocode_OSM(to)
from_point = c(-1.54408, 53.79360)
to_point = c(-1.54802, 53.79618)
# save result from the API call to journey.json
# res_json = journey(from_point, to_point, silent = FALSE, save_raw = TRUE)
# jsonlite::write_json(res_json, "inst/extdata/journey.json")
f = system.file(package = "cyclestreets", "extdata/journey.json")
obj = jsonlite::read_json(f, simplifyVector = TRUE)
rsf = json2sf_cs(obj, cols = c("distances"))
names(rsf)
rsf
rsf2 = json2sf_cs(obj, cols = NULL, cols_extra = NULL)
names(rsf2)
# stplanr::line2points(rsf) extract start and end points
sf:::plot.sf(rsf)
json2sf_cs(obj, cols = c("time", "busynance", "elevations"))
json2sf_cs(obj, cols = c("distances"), smooth_gradient = TRUE,
           gradient_cutoff = 0.05, distance_cutoff = 50)
```

ltns

Download data on 'Low Traffic Neighbourhoods' or 'rat runs' from CycleStreets

Description

R interface to the CycleStreets.net LTN. See ltn API docs and an article on the methods for further details: https://www.cyclestreets.org/news/2021/07/25/mapping-ltns/

Usage

```r
ltns(bb, pat = Sys.getenv("CYCLESTREETS"))
```

Arguments

- **bb**: An sf or 'bounding box' like object
- **pat**: The API key used. By default this uses Sys.getenv("CYCLESTREETS").

Examples

```r
## Not run:
bb <- "0.101131,52.195807,0.170288,52.209719"
ltn_data <- ltls(bb)
plot(ltn_data)
bb <- stplanr::routes_fast_sf
ltn_data <- ltls(bb)
```
### smooth_with_cutoffs

Identify and smooth-out anomalous gradient values

#### Description

When distance_cutoff and gradient_cutoff thresholds are both broken for route segments, this function treats them as anomalous and sets the offending gradient values to the mean of the n segments closest to (in front of and behind) the offending segment.

#### Usage

```r
smooth_with_cutoffs(
  gradient_segment,
  elevation_change,
  distances,
  distance_cutoff = 50,
  gradient_cutoff = 0.1,
  n = 3
)
```

#### Arguments

- `gradient_segment`: The gradient for each segment from CycleStreets.net
- `elevation_change`: The difference between the maximum and minimum elevations within each segment
- `distances`: The distance of each segment
- `distance_cutoff`: Distance (m) used to identify anomalous gradients
- `gradient_cutoff`: Gradient (%, e.g. 0.1 being 10%) used to identify anomalous gradients
- `n`: The number of segments to use to smooth anomalous gradients. The default is 3, meaning segments directly before, after and including the offending segment.

#### Examples

```r
f = system.file(package = "cyclestreets", "extdata/journey.json")
obj = jsonlite::read_json(f, simplifyVector = TRUE)
rsf = json2sf_cs(obj, cols = c("distances"))
rsf$gradient_segment
rsf$elevation_change
rsf$distances
```
smooth_with_cutoffs(rsf$gradient_segment, rsf$elevation_change, rsf$distances)
smooth_with_cutoffs(rsf$gradient_segment, rsf$elevation_change, rsf$distances, 20, 0.05)
smooth_with_cutoffs(rsf$gradient_segment, rsf$elevation_change, rsf$distances, 200, 0.02)
smooth_with_cutoffs(rsf$gradient_segment, rsf$elevation_change, rsf$distances, 200, 0.02, n = 5)
Index

* datasets
  cyclestreets_column_names, 2

cyclestreets_column_names, 2

journey, 2

json2sf_cs, 5

ltns, 6

smooth_with_cutoffs, 7