Package 'cyclestreets'

February 17, 2023

Type Package

Title Cycle Routing and Data for Cycling Advocacy

Version 0.6.0

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

URL https://rpackage.cyclestreets.net/, https://github.com/cyclestreets/cyclestreets-r

BugReports https://github.com/cyclestreets/cyclestreets-r/issues

Depends R (>= 3.6.0)

Imports checkmate, curl, dplyr, geodist, geojsonsf, httr, jsonlite, magrittr, progressr, purrr, R.utils, RcppSimdJson, sf, stringr

Suggests covr, od, stplanr

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

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>), Malcolm Morgan [ctb] (<https://orcid.org/0000-0002-9488-9183>)

Maintainer Robin Lovelace <rob00x@gmail.com>

Repository CRAN

Date/Publication 2023-02-17 09:30:06 UTC

R topics documented:

`batch` *Interface to CycleStreets Batch Routing API*

Description

Note: set `CYCLESTREETS_BATCH` and `CYCLESTREETS_PW` environment variables, e.g. with `usethis::edit_r_environ()` before trying this.

Usage

```
batch(
  desire_lines,
  directory = tempdir(),
  wait_time = NULL,
  name = "test batch",
  serverId = 21,
  strategies = "quietest",
  bothDirections = 0,
  minDistance = 50,
  maxDistance = 5000,
  filename = "test",
  includeJsonOutput = 1,
  emailOnCompletion = "you@example.com",
  username = "yourname",
  password = Sys.getenv("CYCLESTREETS_PW"),
  base_url = "https://api.cyclestreets.net/v2/batchroutes.createjob",
  id = NULL,
  pat = Sys.getenv("CYCLESTREETS_BATCH"),
  silent = TRUE
)
```

Arguments

desire_lines	Geographic desire lines representing origin-destination data
directory	Where to save the data? tempdir() by default
wait_time	How long to wait before getting the data in seconds? NULL by default, meaning it will be calculated by the private function wait_s().
name	The name of the batch routing job for CycleStreets
serverId	The server ID to use (21 by default)
strategies	Route plan types, e.g. "fastest"
bothDirections	int (1|0) Whether to plan in both directions, i.e. A-B as well as B-A. 0, meaning only one way routes, is the default in the R default.
minDistance	Min Euclidean distance of routes to be calculated
maxDistance	Maximum Euclidean distance of routes to be calculated
filename	Character string
includeJsonOutput	int (1|0) Whether to include a column in the resulting CSV data giving the full JSON output from the API, rather than just summary information like distance and time.
emailOnCompletion	Email on completion?
username	string Your CycleStreets account username. In due course this will be replaced with an OAuth token.
password	string Your CycleStreets account password. You can set it with Sys.setenv(CYCLESTREETS_PW="xxxx
base_url	The base url from which to construct API requests (with default set to main server)
id	int Batch job ID, as returned from batchroutes.createjob. action string (start|pause|continue|terminate) Action to take. Available actions are: start: Start (open) job pause: Pause job continue: Continue (re-open) job terminate: Terminate job and delete data
pat	The API key used. By default this uses Sys.getenv("CYCLESTREETS").
silent	Logical (default is FALSE). TRUE hides request sent.

Examples

```
if(FALSE) {
library(sf)
desire_lines = od::od_to_sf(od::od_data_df, od::od_data_zones)[4:5, 1:3]
u = paste0("https://github.com/cyclestreets/cyclestreets-r/",
  "releases/download/v0.5.3/od-longford-10-test.Rds")
desire_lines = readRDS(url(u))
routes = batch(desire_lines, username = "robinlovelace")
names(routes)
plot(routes$geometry)
plot(desire_lines$geometry, add = TRUE, col = "red")
routes = batch(desire_lines, username = "robinlovelace", wait_time = 5)
}
```

cyclestreets_column_names *Prices of 50,000 round cut diamonds.*

Description

Variables provided by CycleStreets in their journey data

Usage

```
cyclestreets_column_names
```

Format

An object of class `character` of length 44.

Source

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.

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", "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,
  warnNA = FALSE
)
```

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/cyclestreet
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 gradents.
warnNA	Logical should NA warning be given? 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:

```
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", "plan", "note", "length", "quietness",
"west", "south", "east", "north", "leaving", "arriving", "grammesCO2saved",
"calories", "edition", "geometry")
```

See www.cyclestreets.net/help/journey/howitworks/ for details on how these are calculated.

See Also

json2sf_cs

Examples

```
## 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)
```

journey2 *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.

Usage

```
journey2(
  fromPlace = NA,
  toPlace = NA,
  id = NULL,
  plan = "fastest",
  pat = NULL,
  base_url = "https://www.cyclestreets.net",
  host_con = 1,
  reporterrors = TRUE,
  segments = FALSE
)
```

Arguments

fromPlace	sf points, matrix, or vector of lng/lat coordinates
toPlace	sf points, matrix, or vector of lng/lat coordinates
id	a character ID value to be attached to the results
plan	Text strong of either "fastest" (default), "quietest" or "balanced"
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)
host_con	number of threads to use passed to curl::new_pool
reporterrors	Boolean value (TRUE/FALSE) indicating if cyclestreets (TRUE by default). should report errors (FALSE by default).
segments	Logical, if true route segments returned otherwise whole routes

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:

```
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.

See www.cyclestreets.net/help/journey/howitworks/ for details on how these are calculated.

See Also

json2sf_cs

Examples

```
## 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)
r2 = journey2(from, to, segments = TRUE)
# waldo::compare(r1, r2) # see differences
sum(sf::st_length(r1))
sum(sf::st_length(r2))
# waldo::compare(sum(sf::st_length(r1)), sum(sf::st_length(r2)))
# waldo::compare(names(r1), names(r2))
# waldo::compare(r1[1, ], r2[1, ])
r1[1:2, ]
r2[1:2, ]
r1$grammesCO2saved
r1$calories

## End(Not run)
```

---

json2sf_cs *Convert output from CycleStreets.net into sf object*

Description

Convert output from CycleStreets.net into sf object

Usage

```
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,
  warnNA = FALSE
)
```

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/cyclestreet
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 gradents.
warnNA	Logical should NA warning be given? The default is 3, meaning segments directly before, after and including the offending segment.

Examples

```
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

```
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

```
## Not run:
bb = "0.101131,52.195807,0.170288,52.209719"
ltn_data = ltns(bb)
plot(ltn_data)
bb = stplanr::routes_fast_sf
ltn_data = ltns(bb)
plot(ltn_data)

## End(Not run)
```

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

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

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 gradents.
warnNA	Logical should NA warning be given? The default is 3, meaning segments directly before, after and including the offending segment.

Examples

```
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)
```

---

ways *Download data on 'Ways' with cyclability (quietness) ratings*

Description

R interface to the CycleStreets.net LTN. See API docs.

Usage

```
ways(
  bb,
  pat = Sys.getenv("CYCLESTREETS"),
  base_url = "https://api.cyclestreets.net/v2/mapdata?",
  limit = 400,
  types = "way",
  wayFields =
   "name,ridingSurface,id,cyclableText,quietness,speedMph,speedKmph,pause,color",
  zoom = 16
)
```

Arguments

bb	An sf or 'bounding box' like object
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)
limit	Maximum number of features to return
types	The type of way to get. Default: "way".
wayFields	Which attributes of the ways to return?
zoom	Zoom level

Examples

```
## Not run:

u_test = paste0("https://api.cyclestreets.net/v2/mapdata?key=c047ed46f7b50b1x",
  "&limit=400&types=way&wayFields=name,ridingSurface,id,cyclableText,",
  "quietness,speedMph,speedKmph,pause,color&zoom=16&",
  "bbox=-9.160863,38.754642,-9.150128,38.75764")
# ways_test = sf::read_sf(u_test)
bb = "0.101131,52.195807,0.170288,52.209719"
bb = "-9.160863,38.754642,-9.150128,38.75764"
way_data = ways(bb)
plot(way_data)
bb = stplanr::routes_fast_sf
way_data = ways(bb)
plot(way_data)

## End(Not run)
```

Index