Package ‘r5r’

July 2, 2021

Type Package
Title Rapid Realistic Routing with 'R5'
Version 0.5.0
Description Rapid realistic routing on multimodal transport networks (walk, bike, public transport and car) using 'R5', the Rapid Realistic Routing on Real-world and Reimagined networks
<https://github.com/conveyal/r5>. The package allows users to generate detailed routing analysis or calculate travel time matrices using seamless parallel computing on top of the R5 Java machine.

Date 2021-07-02
URL https://github.com/ipeaGIT/r5r
BugReports https://github.com/ipeaGIT/r5r/issues
License MIT + file LICENSE
Encoding UTF-8
Depends R (>= 3.6)
Suggests akima, covr, dplyr, ggplot2 (>= 3.3.1), knitr, mapview, markdown, rgdal, testthat
Imports checkmate, curl, data.table, httr, methods, jdx (>= 0.1.4), raster, rJava (>= 0.9-10), sf (>= 0.9-3), sfheaders, utils
SystemRequirements Java JDK (>= 11.0)
RoxygenNote 7.1.1
VignetteBuilder knitr
NeedsCompilation no
Author Marcus Saraiva [aut] (<https://orcid.org/0000-0001-6218-2338>), Rafael H. M. Pereira [aut, cre] (<https://orcid.org/0000-0003-2125-7465>), Daniel Herszenhut [aut] (<https://orcid.org/0000-0001-8066-1105>), Carlos Kaue Vieira Braga [aut] (<https://orcid.org/0000-0002-6104-7297>), Matthew Wigginton Conway [aut] (<https://orcid.org/0000-0002-1210-2982>), Ipea - Institute for Applied Economic Research [cph, fnd]
accessibility

Calculate access to opportunities

Description

Fast computation of access to opportunities given a selected decay function. See details for the available decay functions.
accessibility

Usage

accessibility(
  r5r_core,
  origins, destinations,
  opportunities_colname = "opportunities",
  mode = "WALK",
  mode_egress = "WALK",
  departure_datetime = Sys.time(),
  time_window = 1L,
  percentiles = 50L,
  decay_function = "step",
  cutoffs = 30L,
  decay_value = 1,
  max_walk_dist = Inf,
  max_bike_dist = Inf,
  max_trip_duration = 120L,
  walk_speed = 3.6,
  bike_speed = 12,
  max_rides = 3,
  max_lts = 2,
  n_threads = Inf,
  verbose = TRUE
)

Arguments

r5r_core a rJava object to connect with R5 routing engine
origins, destinations a spatial sf POINT object, or a data.frame containing the columns 'id', 'lon', 'lat'
opportunities_colname string. The column name in the destinations input that tells the number of opportunities in each location. Defaults to "opportunities".
mode string. Transport modes allowed for the trips. Defaults to "WALK". See details for other options.
mode_egress string. Transport mode used after egress from public transport. It can be either 'WALK', 'BICYCLE', or 'CAR'. Defaults to "WALK".
departure_datetime POSIXct object. If working with public transport networks, please check calendar.txt within the GTFS file for valid dates.
time_window numeric. Time window in minutes for which r5r will calculate travel times departing each minute. When using frequency-based GTFS files, 5 Monte Carlo simulations will be run for each minute in the time window. See details for further information.
percentiles numeric vector. Defaults to '50', returning the accessibility value for the median travel time computed for a given time_window. If a numeric vector is
accessibility

passed, for example c(25, 50, 75), the function will return accessibility estimates for each percentile, by travel time cutoff. Only the first 5 cut points of the percentiles are considered. For more details, see R5 documentation at 'https://docs.conveyal.com/analysis/methodology#accounting-for-variability’

decay_function string. Choice of one of the following decay functions: 'step', 'exponential', 'fixed_exponential', 'linear', and 'logistic'. Defaults to 'step', which yields cumulative opportunities accessibility metrics. More info in details.
cutoffs numeric. Cutoff times in minutes for calculating cumulative opportunities accessibility when using the 'step decay function’. This parameter has different effects for each of the other decay functions: it indicates the 'median' (or inflection point) of the decay curves in the 'logistic' and 'linear' functions, and the 'half-life' in the 'exponential' function. It has no effect when using the 'fixed exponential' function.
decay_value numeric. Extra parameter to be passed to the selected decay_function.
max_walk_dist numeric. Maximum walking distance (in meters) for the whole trip. Defaults to no restrictions on walking, as long as max_trip_duration is respected.
max_bike_dist numeric. Maximum cycling distance (in meters) for the whole trip. Defaults to no restrictions on cycling, as long as max_trip_duration is respected.
max_trip_duration numeric. Maximum trip duration in minutes. Defaults to 120 minutes (2 hours).
walk_speed numeric. Average walk speed in km/h. Defaults to 3.6 km/h.
bike_speed numeric. Average cycling speed in km/h. Defaults to 12 km/h.
max_rides numeric. The max number of public transport rides allowed in the same trip. Defaults to 3.
max_lts numeric (between 1 and 4). The maximum level of traffic stress that cyclists will tolerate. A value of 1 means cyclists will only travel through the quietest streets, while a value of 4 indicates cyclists can travel through any road. Defaults to 2. See details for more information.
n_threads numeric. The number of threads to use in parallel computing. Defaults to use all available threads (Inf).
verbose logical. TRUE to show detailed output messages (the default). If verbose is set to FALSE, r5r prints a progress counter and eventual ERROR messages. Setting verbose to FALSE imposes a small penalty for computation efficiency.

Value

A data.table with accessibility estimates for all origin points, by a given transport mode, and per travel time cutoff and percentile.

Decay functions:

R5 allows for multiple decay functions. More info at https://docs.conveyal.com/learn-more/decay-functions The options include:

Step step (cumulative opportunities):
A binary decay function used to calculate cumulative opportunities metrics.
Logistic CDF logistic:
This is the logistic function, i.e. the cumulative distribution function of the logistic distribution, expressed such that its parameters are the median (inflection point) and standard deviation. This function applies a sigmoid rolloff that has a convenient relationship to discrete choice theory. Its parameters can be set to reflect a whole population’s tolerance for making trips with different travel times. The function’s value represents the probability that a randomly chosen member of the population would accept making a trip, given its duration. Opportunities are then weighted by how likely it is a person would consider them "reachable".

**calibration:**
The median parameter is controlled by the cutoff parameter, leaving only the standard deviation to configure through the decay_value parameter.

Fixed Exponential fixed_exponential:
This function is of the form e^{-Lt} where L is a single fixed decay constant in the range (0, 1). It is constrained to be positive to ensure weights decrease (rather than grow) with increasing travel time.

**calibration:**
This function is controlled exclusively by the L constant, given by the decay_value parameter. Values provided in cutoffs are ignored.

Half-life Exponential Decay exponential:
This is similar to the fixed-exponential option above, but in this case the decay parameter is inferred from the cutoffs parameter values, which is treated as the half-life of the decay.

Linear linear:
This is a simple, vaguely sigmoid option, which may be useful when you have a sense of a maximum travel time that would be tolerated by any traveler, and a minimum time below which all travel is perceived to be equally easy.

**calibration:**
The transition region is transposable and symmetric around the cutoffs parameter values, taking decay_value minutes to taper down from one to zero.

Transport modes:

R5 allows for multiple combinations of transport modes. The options include:

**Transit modes:**
TRAM, SUBWAY, RAIL, BUS, FERRY, CABLE_CAR, GONDOLA, FUNICULAR. The option 'TRANSIT' automatically considers all public transport modes available.

**Non transit modes:**
WALK, BICYCLE, CAR, BICYCLE_RENT, CAR_PARK

max_lts, Maximum Level of Traffic Stress:

When cycling is enabled in R5, setting max_lts will allow cycling only on streets with a given level of danger/stress. Setting max_lts to 1, for example, will allow cycling only on separated bicycle infrastructure or low-traffic streets; routing will revert to walking when traversing any links with LTS exceeding 1. Setting max_lts to 3 will allow cycling on links with LTS 1, 2, or 3.
The default methodology for assigning LTS values to network edges is based on commonly tagged attributes of OSM ways. See more info about LTS at [https://docs.conveyal.com/learn-more/traffic-stress](https://docs.conveyal.com/learn-more/traffic-stress). In summary:

- **LTS 1**: Tolerable for children. This includes low-speed, low-volume streets, as well as those with separated bicycle facilities (such as parking-protected lanes or cycle tracks).
- **LTS 2**: Tolerable for the mainstream adult population. This includes streets where cyclists have dedicated lanes and only have to interact with traffic at formal crossing.
- **LTS 3**: Tolerable for “enthused and confident” cyclists. This includes streets which may involve close proximity to moderate- or high-speed vehicular traffic.
- **LTS 4**: Tolerable for only “strong and fearless” cyclists. This includes streets where cyclists are required to mix with moderate- to high-speed vehicular traffic.

**Routing algorithm:**

The `accessibility()` function uses an R5-specific extension to the RAPTOR routing algorithm (see Conway et al., 2017). This RAPTOR extension uses a systematic sample of one departure per minute over the time window set by the user in the ‘time_window’ parameter. A detailed description of base RAPTOR can be found in Delling et al (2015).


**See Also**

Other routing: `detailed_itineraries()`, `travel_time_matrix()`

**Examples**

```r
if (interactive()) {
  library(r5r)

  # build transport network
data_path <- system.file("extdata/poa", package = "r5r")
r5r_core <- setup_r5(data_path = data_path)

  # load origin/destination points
points <- read.csv(file.path(data_path, "poa_hexgrid.csv"))

  access <- accessibility(r5r_core,
                           origins = points,
                           destinations = points,
                           opportunities_colname = "schools",
                           mode = "WALK",
                           cutoffs = c(25, 30),
                           max_trip_duration = 30,
                           verbose = FALSE)
```

```
apply_elevation

stop_r5(r5r_core)
}

apply_elevation  Apply elevation to street network

Description
Loads a Digital Elevation Model (DEM) from a raster file and weights the street network for walking and cycling according to the terrain’s slopes.

Usage
apply_elevation(r5r_core, raster_files)

Arguments
- r5r_core: a rJava object to connect with R5 routing engine.
- raster_files: string. Path to raster files containing the study area’s topography. If a list is provided, all the rasters are automatically merged.

Value
No return value, called for side effects.

See Also
Other elevation support functions: tobler_hiking()

assert_decay_function  Assert decay function and parameter values

Description
Assert decay function and parameter values.

Usage
assert_decay_function(decay_function, decay_value)

Arguments
- decay_function: Name of decay function.
- decay_value: Value of decay parameter.
**Value**

A list with the validated decay function and parameter value.

**See Also**

Other support functions: `assert_points_input()`, `check_connection()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_rides()`, `set_max_street_time()`, `set_n_threads()`, `set_speed()`, `set_suboptimal_minutes()`, `setVerbose()`, `stop_r5()`, `street_network_to_sf()`

---

**assert_points_input**  Assert class of origin and destination inputs and the type of its columns

**Description**

Assert class of origin and destination inputs and the type of its columns

**Usage**

```r
assert_points_input(df, name)
```

**Arguments**

- **df**  Any object.
- **name**  Object name.

**Value**

A data.frame with columns `id`, `lon` and `lat`.

**See Also**

Other support functions: `assert_decay_function()`, `check_connection()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_rides()`, `set_max_street_time()`, `set_n_threads()`, `set_speed()`, `set_suboptimal_minutes()`, `setVerbose()`, `stop_r5()`, `street_network_to_sf()`
**check_connection**  
*Check internet connection with Ipea server*

**Description**  
Checks if there is internet connection to Ipea server to download geobr data.

**Usage**

```r
check_connection(
  file_url = "https://www.ipea.gov.br/geobr/metadata/metadata_gpkg.csv"
)
```

**Arguments**

- **file_url**  
  A string with the file_url address of an geobr dataset

**Value**  
No return value, called for side effects.

**See Also**

Other support functions: `assert_decay_function()`, `assert_points_input()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_rides()`, `set_max_street_time()`, `set_n_threads()`, `set_speed()`, `set_suboptimal_minutes()`, `set_verbose()`, `stop_r5()`, `street_network_to_sf()`

---

**detailed_itineraries**  
*Calculate detailed itineraries between origin destination pairs*

**Description**

Fast computation of (multiple) detailed itineraries between one or many origin destination pairs.

**Usage**

```r
detailed_itineraries(
  r5r_core,  
  origins,  
  destinations,  
  mode = "WALK",  
  mode_egress = "WALK",  
  departure_datetime = Sys.time(),  
  max_walk_dist = Inf,  
  max_bike_dist = Inf,  
  max_trip_duration = 120L,
```

---
walk_speed = 3.6,  
bike_speed = 12,  
max_rides = 3,  
max_lts = 2,  
shortest_path = TRUE,  
n_threads = Inf,  
verbose = TRUE,  
drop_geometry = FALSE
)

Arguments

r5r_core rJava object to connect with R5 routing engine

origins, destinations  
either a spatial sf POINT object or a data.frame containing the columns 'id', 'lon', 'lat'

mode string. Transport modes allowed for the trips. Defaults to "WALK". See details for other options.

mode_egress string. Transport mode used after egress from public transport. It can be either 'WALK', 'BICYCLE', or 'CAR'. Defaults to "WALK". Ignored when public transport is not used.

departure_datetime POSIXct object. If working with public transport networks, please check calendar.txt within the GTFS file for valid dates.

max_walk_dist numeric. Maximum walking distance (in meters) for the whole trip. Defaults to no restrictions on walking, as long as max_trip_duration is respected.

max_bike_dist numeric. Maximum cycling distance (in meters) for the whole trip. Defaults to no restrictions on cycling, as long as max_trip_duration is respected.

max_trip_duration numeric. Maximum trip duration in minutes. Defaults to 120 minutes (2 hours).

walk_speed numeric. Average walk speed in km/h. Defaults to 3.6 km/h.

bike_speed numeric. Average cycling speed in km/h. Defaults to 12 km/h.

max_rides numeric. The max number of public transport rides allowed in the same trip. Defaults to 3.

max_lts numeric (between 1 and 4). The maximum level of traffic stress that cyclists will tolerate. A value of 1 means cyclists will only travel through the quietest streets, while a value of 4 indicates cyclists can travel through any road. Defaults to 2. See details for more information.

shortest_path logical. Whether the function should only return the fastest route alternative (the default) or multiple alternatives.

n_threads numeric. The number of threads to use in parallel computing. Defaults to use all available threads (Inf).

verbose logical. TRUE to show detailed output messages (the default). If verbose is set to FALSE, r5r prints a progress counter and eventual ERROR messages. Setting verbose to FALSE imposes a small penalty for computation efficiency.
detailed_itineraries

drop_geometry logical. Indicates whether R5 should drop segment’s geometry column. It can be helpful for saving memory.

Value

A LINESTRING sf with detailed information about the itineraries between specified origins and destinations. Distances are in meters and travel times are in minutes.

Transport modes:

R5 allows for multiple combinations of transport modes. The options include:

Transit modes:
TRAM, SUBWAY, RAIL, BUS, FERRY, CABLE_CAR, GONDOLA, FUNICULAR. The option "TRANSIT" automatically considers all public transport modes available.

Non transit modes:
WALK, BICYCLE, CAR, BICYCLE_RENT, CAR_PARK

max_lts, Maximum Level of Traffic Stress:

When cycling is enabled in R5, setting max_lts will allow cycling only on streets with a given level of danger/stress. Setting max_lts to 1, for example, will allow cycling only on separated bicycle infrastructure or low-traffic streets; routing will revert to walking when traversing any links with LTS exceeding 1. Setting max_lts to 3 will allow cycling on links with LTS 1, 2, or 3.

The default methodology for assigning LTS values to network edges is based on commonly tagged attributes of OSM ways. See more info about LTS at https://docs.conveyal.com/learn-more/traffic-stress. In summary:

- LTS 1: Tolerable for children. This includes low-speed, low-volume streets, as well as those with separated bicycle facilities (such as parking-protected lanes or cycle tracks).
- LTS 2: Tolerable for the mainstream adult population. This includes streets where cyclists have dedicated lanes and only have to interact with traffic at formal crossing.
- LTS 3: Tolerable for “enthused and confident” cyclists. This includes streets which may involve close proximity to moderate- or high-speed vehicular traffic.
- LTS 4: Tolerable for only “strong and fearless” cyclists. This includes streets where cyclists are required to mix with moderate- to high-speed vehicular traffic.

Routing algorithm:

The detailed_itineraries function uses an R5-specific extension to the McRAPTOR routing algorithm to find paths that are optimal or less than optimal, with some heuristics around multiple access modes, riding the same patterns, etc. The specific extension to McRAPTOR to do suboptimal path routing are not documented yet, but a detailed description of base McRAPTOR can be found in Delling et al (2015).

download_metadata

Description
Support function to download metadata internally used in r5r

Usage
download_metadata()

Value
A data.frame with url address of r5r Jar files

See Also
Other support functions: assert_decay_function(), assert_points_input(), check_connection(), find_snap(), posix_to_string(), select_mode(), set_max_lts(), set_max_rides(), set_max_street_time(), set_n_threads(), set_speed(), set_suboptimal_minutes(), set_verbose(), stop_r5(), street_network_to_sf()
download_r5

Description

Download a compiled JAR file of R5 and saves it locally. The JAR file is saved within the package directory. The package uses a compilation of R5 tailored for the purposes of r5r that keeps R5’s essential features. Source code available at https://github.com/ipeaGIT/r5r.

Usage

download_r5(
  version = "6.2.0",
  quiet = FALSE,
  force_update = FALSE,
  temp_dir = FALSE
)

Arguments

version character string with the version of R5 to be downloaded. Defaults to latest version '1.0'.
quiet logical, passed to download.file. Defaults to FALSE
force_update logical, Replaces the jar file stored locally with a new one. Defaults to FALSE.
temp_dir logical, whether the R5 Jar file should be saved in temporary directory. Defaults to FALSE

Value

A jar file is saved locally in the r5r package directory

See Also

Other setup: setup_r5()

Examples

if (interactive()) {

  library(r5r)

  download_r5(version = "6.2", temp_dir = TRUE)
}


find_snap

Find snapped locations of input points on street network

Description

R5 tries to snap origin and destination points to the street network in two rounds. First, it uses a search radius of 300 meters. If the first round is unsuccessful, then R5 expands the search radius to 1.6 km. Points that aren’t linked to the street network after those two rounds are returned with NA coordinates and found = FALSE. Please note that the location of the snapped points depends on the transport mode set by the user.

Usage

find_snap(r5r_core, points, mode = "WALK")

Arguments

- r5r_core: a rJava object to connect with R5 routing engine
- points: a spatial sf POINT object, or a data.frame containing the columns 'id', 'lon', 'lat'
- mode: string. Defaults to "WALK", also allows "BICYCLE", and "CAR".

Value

A data.table with the original points as well as their respective snapped coordinates on the street network and the Euclidean distance between original points and their respective snapped location. Points that could not be snapped show NA coordinates and found = FALSE.

See Also

Other support functions: assert_decay_function(), assert_points_input(), check_connection(), download_metadata(), posix_to_string(), select_mode(), set_max_lts(), set_max_rides(), set_max_street_time(), set_n_threads(), set_speed(), set_suboptimal_minutes(), set_verbose(), stop_r5(), street_network_to_sf()

Examples

if (interactive()) {

library(r5r)

# build transport network
path <- system.file("extdata/spo", package = "r5r")
r5r_core <- setup_r5(data_path = path)

# load origin/destination points
points <- read.csv(file.path(path, "spo_hexgrid.csv"))
# find where origin or destination points are snapped
snap_df <- find_snap(r5r_core, 
    points = points, 
    mode = 'CAR')

stop_r5(r5r_core)
}

---

### posix_to_string

Generate date and departure time strings from POSIXct

**Description**

Generate date and departure time strings from POSIXct

**Usage**

```r
posix_to_string(datetime)
```

**Arguments**

- `datetime`: An object of POSIXct class.

**Value**

A list with the date and time of the trip departure as characters

A list with 'date' and 'departure_time' names.

**See Also**

Other support functions: assert_decay_function(), assert_points_input(), check_connection(),
download_metadata(), find_snap(), select_mode(), set_max_lts(), set_max_rides(), set_max_street_time(),
set_n_threads(), set_speed(), set_suboptimal_minutes(), set_verbose(), stop_r5(), street_network_to_sf()

---

### select_mode

Select transport mode

**Description**

Select transport mode

**Usage**

```r
select_mode(mode, mode_egress)
```
setup_r5

Create transport network used for routing in R5

Description

Combine data inputs in a directory to build a multimodal transport network used for routing in R5. The directory must contain at least one street network file (in .pbf format). One or more public transport data sets (in GTFS.zip format) are optional. If there is more than one GTFS file in the directory, both files will be merged. If there is already a ‘network.dat’ file in the directory the function will simply read it and load it to memory.

Usage

setup_r5(
  data_path,
  version = "6.2.0",
  verbose = TRUE,
  temp_dir = FALSE,
  use_elevation = FALSE
)

Arguments

data_path character string, the directory where data inputs are stored and where the built network.dat will be saved.
version character string, the version of R5 to be used. Defaults to latest version ‘6.2.0’.
verbose logical, TRUE to show detailed output messages (Default) or FALSE to show only eventual ERROR and WARNING messages.
temp_dir logical, whether the R5 Jar file should be saved in temporary directory. Defaults to FALSE
use_elevation boolean. If TRUE, load any tif files containing elevation found in the data_path folder and calculate impedances for walking and cycling based on street slopes.
set_max_lts

Value
An rJava object to connect with R5 routing engine

See Also
Other setup: download_r5()

Examples

```r
if (interactive()) {
    library(r5r)

    # directory with street network and gtfs files
    path <- system.file("extdata/poa", package = "r5r")

    r5r_core <- setup_r5(data_path = path, temp_dir = TRUE)
}
```

---

set_max_lts  

Set max Level of Transit Stress (LTS)

Description
Set max Level of Transit Stress (LTS)

Usage

```r
set_max_lts(r5r_core, max_lts)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r5r_core</td>
<td>rJava object to connect with R5 routing engine</td>
</tr>
<tr>
<td>max_lts</td>
<td>numeric (between 1 and 4). The maximum level of traffic stress that cyclists will tolerate. A value of 1 means cyclists will only travel through the quietest streets, while a value of 4 indicates cyclists can travel through any road.</td>
</tr>
</tbody>
</table>

Value
No return value, called for side effects.

See Also
Other support functions: assert_decay_function(), assert_points_input(), check_connection(), download_metadata(), find_snap(), posix_to_string(), select_mode(), set_max_rides(), set_max_street_time(), set_n_threads(), set_speed(), set_suboptimal_minutes(), set_verbose(), stop_r5(), street_network_to_sf()
set_max_rides

**Set max number of transfers**

**Description**

Set maxTransfers parameter in R5.

**Usage**

```
set_max_rides(r5r_core, max_rides)
```

**Arguments**

- `r5r_core`: rJava object to connect with R5 routing engine.
- `max_rides`: numeric. The max number of public transport rides allowed in the same trip. Passed from routing function.

**Value**

No return value, called for side effects.

**See Also**

Other support functions: `assert_decay_function()`, `assert_points_input()`, `check_connection()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_street_time()`, `set_n_threads()`, `set_speed()`, `set_suboptimal_minutes()`, `set_verbose()`, `stop_r5()`, `street_network_to_sf()`

---

set_max_street_time

**Set max street time**

**Description**

Set max street time

**Usage**

```
set_max_street_time(max_walk_dist, walk_speed, max_trip_duration)
```

**Arguments**

- `max_walk_dist`: numeric, Maximum walking distance (in meters) for the whole trip. Passed from routing functions.
- `walk_speed`: numeric, Average walk speed in Km/h. Defaults to 3.6 Km/h. Passed from routing functions.
- `max_trip_duration`: numeric, Maximum trip duration in seconds. Defaults to 120 minutes (2 hours). Passed from routing functions.
**set_n_threads**

**Value**

An integer representing the maximum number of minutes walking

**See Also**

Other support functions: `assert_decay_function()`, `assert_points_input()`, `check_connection()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_rides()`, `set_n_threads()`, `set_speed()`, `set_suboptimal_minutes()`, `setVerbose()`, `stop_r5()`, `street_network_to_sf()`

---

### set_n_threads

#### Description

Sets number of threads to be used by the r5r.jar.

#### Usage

```r
set_n_threads(r5r_core, n_threads)
```

#### Arguments

- **r5r_core**: a rJava object to connect with R5 routing engine
- **n_threads**: Any object.

#### Value

No return value, called for side effects.

#### See Also

Other support functions: `assert_decay_function()`, `assert_points_input()`, `check_connection()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_rides()`, `set_max_street_time()`, `set_speed()`, `set_suboptimal_minutes()`, `setVerbose()`, `stop_r5()`, `street_network_to_sf()`
Description

This function receives the walk and bike 'speed' inputs in Km/h from routing functions above and converts them to meters per second, which is then used to set these speed profiles in r5r JAR.

Usage

```r
set_speed(r5r_core, speed, mode)
```

Arguments

- `r5r_core`: a rJava object to connect with R5 routing engine
- `speed`: A numeric representing the speed in km/h.
- `mode`: Either "bike" or "walk".

Value

No return value, called for side effects.

See Also

Other support functions: assert_decay_function(), assert_points_input(), check_connection(), download_metadata(), find_snap(), posix_to_string(), select_mode(), set_max_lts(), set_max_rides(), set_max_street_time(), set_n_threads(), set_suboptimal_minutes(), set_verbose(), stop_r5(), street_network_to_sf()
**set_verbose**

**Arguments**

- `r5r_core` rJava object to connect with R5 routing engine
- `suboptimal_minutes` numeric. The number of suboptimal minutes in a public transport point-to-point query. From R5’s documentation: This parameter compensates for the fact that GTFS does not contain information about schedule deviation (lateness). The min-max travel time range for some trains is zero, since the trips are reported to always have the same timings in the schedule. Such an option does not overlap (temporally) its alternatives, and is too easily eliminated by an alternative that is only marginally better. We want to effectively push the max travel time of alternatives out a bit to account for the fact that they don’t always run on schedule.

**Value**

No return value, called for side effects.

**See Also**

Other support functions: `assert_decay_function()`, `assert_points_input()`, `check_connection()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_rides()`, `set_max_street_time()`, `set_n_threads()`, `set_speed()`, `set_verbose()`, `stop_r5()`, `street_network_to_sf()`

---

**set_verbose**  
*Set verbose argument*

**Description**

Set verbose argument

**Usage**

`set_verbose(r5r_core, verbose)`

**Arguments**

- `r5r_core` a rJava object to connect with R5 routing engine
- `verbose` logical, passed from function above

**Value**

No return value, called for side effects.
**Stop running r5r core**

**Description**

Stops running r5r cores.

**Usage**

`stop_r5(...)`

**Arguments**

`...`  
`r5r_core` objects currently running. By default, if no cores are supplied all running cores are stopped.

**Value**

No return value, called for side effects.

**See Also**

Other support functions: `assert_decay_function()`, `assert_points_input()`, `check_connection()`, `download_metadata()`, `find_snap()`, `posix_to_string()`, `select_mode()`, `set_max_lts()`, `set_max_rides()`, `set_max_street_time()`, `set_n_threads()`, `set_speed()`, `set_suboptimal_minutes()`, `stop_r5()`, `street_network_to_sf()`

**Examples**

```r
if (interactive()) {
  library(r5r)

  path <- system.file("extdata/poa", package = "r5r")

  r5r_core <- setup_r5(data_path = path)

  stop_r5(r5r_core)
}
```
street_network_to_sf

Extract OpenStreetMap network in sf format from a network.dat built with setup_r5

Description

Extract OpenStreetMap network in sf format from a network.dat built with setup_r5

Usage

street_network_to_sf(r5r_core)

Arguments

r5r_core a rJava object, the output from 'r5r::setup_r5()'

Value

A list with two components of a street network in sf format: vertices (POINT) and edges (LINESTRING).

See Also

Other support functions: assert_decay_function(), assert_points_input(), check_connection(), download_metadata(), find_snap(), posix_to_string(), select_mode(), set_max_lts(), set_max_rides(), set_max_street_time(), set_n_threads(), set_speed(), set_suboptimal_minutes(), set_verbose(), stop_r5()

Examples

if (interactive()) {
  library(r5r)

  # build transport network
  path <- system.file("extdata/poa", package = "r5r")
  r5r_core <- setup_r5(data_path = path)

  # extract street network from r5r_core
  street_net <- street_network_to_sf(r5r_core)

  stop_r5(r5r_core)
}
tobler_hiking  
*Tobler’s hiking function*

**Description**

Calculates effect of the topography on walking speeds, using Tobler’s hiking function.

**Usage**

```r
> tobler_hiking(slope)
```

**Arguments**

- `slope` numeric. Terrain’s slope.

**Value**

numeric. Tobler’s weighting factor

**See Also**

Other elevation support functions: `apply_elevation()`

---

travel_time_matrix  
*Calculate travel time matrix between origin destination pairs*

**Description**

Fast computation of travel time estimates between one or multiple origin destination pairs.

**Usage**

```r
> travel_time_matrix(
>   r5r_core,
>   origins,
>   destinations,
>   mode = "WALK",
>   mode_egress = "WALK",
>   departure_datetime = Sys.time(),
>   time_window = 1L,
>   percentiles = 50L,
>   max_walk_dist = Inf,
>   max_bike_dist = Inf,
>   max_trip_duration = 120L,
>   walk_speed = 3.6,
>   bike_speed = 12
> )
```
Arguments

- **r5r_core**: A rJava object to connect with R5 routing engine.
- **origins, destinations**: A spatial sf POINT object, or a data.frame containing the columns 'id', 'lon', 'lat'.
- **mode**: String. Transport modes allowed for the trips. Defaults to "WALK". See details for other options.
- **mode_egress**: String. Transport mode used after egress from public transport. It can be either 'WALK', 'BICYCLE', or 'CAR'. Defaults to "WALK".
- **departure_datetime**: POSIXct object. If working with public transport networks, please check calendar.txt within the GTFS file for valid dates.
- **time_window**: Numeric. Time window in minutes for which r5r will calculate multiple travel time matrices departing each minute. By default, the number of simulations is 5 times the size of 'time_window' set by the user. Defaults window size to '1', the function only considers 5 departure times. This parameter is only used with frequency-based GTFS files. See details for further information.
- **percentiles**: Numeric vector. Defaults to '50', returning the median travel time for a given time_window. If a numeric vector is passed, for example c(25, 50, 75), the function will return additional columns with the travel times within percentiles of trips. For example, if the 25 percentile of trips between A and B is 15 minutes, this means that 25% of all trips taken between A and B within the set time window are shorter than 15 minutes. Only the first 5 cut points of the percentiles are considered. For more details, see R5 documentation at 'https://docs.conveyal.com/analysis/methodology#accounting-for-variability'
- **max_walk_dist**: Numeric. Maximum walking distance (in meters) for the whole trip. Defaults to no restrictions on walking, as long as max_trip_duration is respected.
- **max_bike_dist**: Numeric. Maximum cycling distance (in meters) for the whole trip. Defaults to no restrictions on cycling, as long as max_trip_duration is respected.
- **max_trip_duration**: Numeric. Maximum trip duration in minutes. Defaults to 120 minutes (2 hours).
- **walk_speed**: Numeric. Average walk speed in km/h. Defaults to 3.6 km/h.
- **bike_speed**: Numeric. Average cycling speed in km/h. Defaults to 12 km/h.
- **max_rides**: Numeric. The max number of public transport rides allowed in the same trip. Defaults to 3.
- **max_lts**: Numeric (between 1 and 4). The maximum level of traffic stress that cyclists will tolerate. A value of 1 means cyclists will only travel through the quietest streets, while a value of 4 indicates cyclists can travel through any road. Defaults to 2. See details for more information.
n_threads numeric. The number of threads to use in parallel computing. Defaults to use all available threads (Inf).

verbose logical. TRUE to show detailed output messages (the default). If verbose is set to FALSE, r5r prints a progress counter and eventual ERROR messages. Setting verbose to FALSE imposes a small penalty for computation efficiency.

Value

A data.table with travel time estimates (in minutes) between origin destination pairs by a given transport mode. Note that origins/destinations that were beyond the maximum travel time, and/or origins that were far from the street network are not returned in the data.table.

Transport modes:

R5 allows for multiple combinations of transport modes. The options include:

Transit modes:
TRAM, SUBWAY, RAIL, BUS, FERRY, CABLE_CAR, GONDOLA, FUNICULAR. The option 'TRANSIT' automatically considers all public transport modes available.

Non transit modes:
WALK, BICYCLE, CAR, BICYCLE_RENT, CAR_PARK

max_lts, Maximum Level of Traffic Stress:

When cycling is enabled in R5, setting max_lts will allow cycling only on streets with a given level of danger/stress. Setting max_lts to 1, for example, will allow cycling only on separated bicycle infrastructure or low-traffic streets; routing will revert to walking when traversing any links with LTS exceeding 1. Setting max_lts to 3 will allow cycling on links with LTS 1, 2, or 3.

The default methodology for assigning LTS values to network edges is based on commonly tagged attributes of OSM ways. See more info about LTS at https://docs.conveyal.com/learn-more/traffic-stress. In summary:

- LTS 1: Tolerable for children. This includes low-speed, low-volume streets, as well as those with separated bicycle facilities (such as parking-protected lanes or cycle tracks).
- LTS 2: Tolerable for the mainstream adult population. This includes streets where cyclists have dedicated lanes and only have to interact with traffic at formal crossing.
- LTS 3: Tolerable for “enthused and confident” cyclists. This includes streets which may involve close proximity to moderate- or high-speed vehicular traffic.
- LTS 4: Tolerable for only “strong and fearless” cyclists. This includes streets where cyclists are required to mix with moderate- to high-speed vehicular traffic.

Routing algorithm:

The travel_time_matrix function uses an R5-specific extension to the RAPTOR routing algorithm (see Conway et al., 2017). This RAPTOR extension uses a systematic sample of one departure per minute over the time window set by the user in the ‘time_window’ parameter. A detailed description of base RAPTOR can be found in Delling et al (2015).


See Also

Other routing: `accessibility()`, `detailed_itineraries()`

Examples

```r
if (interactive()) {
  library(r5r)

  # build transport network
  data_path <- system.file("extdata/spo", package = "r5r")
  r5r_core <- setup_r5(data_path = data_path)

  # load origin/destination points
  points <- read.csv(file.path(data_path, "spo_hexgrid.csv"))[1:5,]
  departure_datetime <- as.POSIXct("13-05-2019 14:00:00", format = "%d-%m-%Y %H:%M:%S")

  # estimate travel time matrix
  ttm <- travel_time_matrix(r5r_core, 
                            origins = points, 
                            destinations = points, 
                            mode = c("WALK", "TRANSIT"), 
                            departure_datetime = departure_datetime, 
                            max_walk_dist = Inf, 
                            max_trip_duration = 120L)

  stop_r5(r5r_core)
}
```
Index

* elevation support functions
  apply_elevation, 7
  tobler_hiking, 24
* routing
  accessibility, 2
  detailed_itineraries, 9
  travel_time_matrix, 24
* setup
  download_r5, 13
  setup_r5, 16
* support functions
  assert_decay_function, 7
  assert_points_input, 8
  check_connection, 9
  download_metadata, 12
  find_snap, 14
  posix_to_string, 15
  select_mode, 15
  set_max_lts, 17
  set_max_rides, 18
  set_max_street_time, 18
  set_n_threads, 19
  set_speed, 20
  set_suboptimal_minutes, 20
  set VERBOSE, 21
  stop_r5, 22
  street_network_to_sf, 23
  accessibility, 2, 12, 27
  apply_elevation, 7, 24
  assert_decay_function, 7, 8, 9, 12, 14–23
  assert_points_input, 8, 9, 12, 14–23
  check_connection, 8, 9, 12, 14–23
  detailed_itineraries, 6, 9, 27
  download_metadata, 8, 9, 12, 14–23
  download_r5, 13, 17
  find_snap, 8, 9, 12, 14, 15–23
  posix_to_string, 8, 9, 12, 14, 15, 16–23
  select_mode, 8, 9, 12, 14, 15, 15, 17–23
  set_max_lts, 8, 9, 12, 14–16, 17, 18–23
  set_max_rides, 8, 9, 12, 14–17, 18, 19–23
  set_max_street_time, 8, 9, 12, 14–18, 18,
  19–23
  set_n_threads, 8, 9, 12, 14–19, 19, 20–23
  set_speed, 8, 9, 12, 14–19, 20, 21–23
  set_suboptimal_minutes, 8, 9, 12, 14–20,
  20, 22, 23
  set VERBOSE, 8, 9, 12, 14–21, 21, 22, 23
  setup_r5, 13, 16
  stop_r5, 8, 9, 12, 14–22, 22, 23
  street_network_to_sf, 8, 9, 12, 14–22, 23
  tobler_hiking, 7, 24
  travel_time_matrix, 6, 12, 24