Package: abstr (via r-universe)

September 13, 2024

Type Package

Title R Interface to the A/B Street Transport System Simulation Software

Version 0.4.1

Description Provides functions to convert origin-destination data, represented as straight 'desire lines' in the 'sf' Simple Features class system, into JSON files that can be directly imported into A/B Street <https://www.abstreet.org>, a free and open source tool for simulating urban transport systems and scenarios of change <doi:10.1007/s10109-020-00342-2>.

License Apache License (>= 2)

URL https://github.com/a-b-street/abstr, https://a-b-street.github.io/abstr/

BugReports https://github.com/a-b-street/abstr/issues

Depends R (>= 4.0.0)

Imports jsonlite (>= 1.7.2), lwgeom (>= 0.2.5), magrittr (>= 2.0.1), methods, od (>= 0.3.1), sf (>= 1.0.1), tibble (>= 3.0.6), tidyr (>= 1.1.3)

Suggests knitr, dplyr (>= 1.0.0), rmarkdown, tmap, pct, foreign

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

VignetteBuilder knitr

Repository https://a-b-street.r-universe.dev

RemoteUrl https://github.com/a-b-street/abstr

RemoteRef HEAD

RemoteSha 0ea64af884cabcd9e581a2f6944adf3639b79788

Contents

ab_json . . . 2
ab_save . . . 3
ab_scenario . . . 4
ab_sf . . . 5
ab_time_normal . . . 6
leeds_site_area . . . 7
montlake_buildings . . . 7
montlake_od . . . 8
montlake_zones . . . 9
sao_paulo_activity_df_20 . . . 9
sao_paulo_activity_sf_20 . . . 10

Index **12**

`ab_json` *Convert geographic ('sf') representation of OD data to 'JSON list' structure*

Description

This function takes outputs from `ab_scenario()` and returns a list that can be saved as a JSON file for import into A/B Street.

Usage

```
ab_json(
  desire_lines,
  mode_column = NULL,
  time_fun = ab_time_normal,
  scenario_name = "test",
  default_purpose = "Work",
  ...
)
```

Arguments

`desire_lines`	OD data represented as geographic lines created by `ab_scenario()`.
`mode_column`	The column name in the desire lines data that contains the mode of transport. `"mode_baseline"` by default.
`time_fun`	The function used to calculate departure times. `ab_time_normal()` by default.
`scenario_name`	The name of the scenario to appear in A/B Street. The default value is `"test"`, which generates a message to tell you to think of a more imaginative scenario name!

default_purpose
: In case a purpose column is not present in the input, or there are missing values in the purpose column, this argument sets the default, fallback purpose, as "Work" by default, reflecting the prevalence of work-based data and thinking in transport models.

...
: Additional arguments to pass to ab_json()

Details

Note: the departure time in seconds is multiplied by 10000 on conversion to a .json list object for compatibility with the A/B Street schema.

Value

A list that can be saved as a JSON file with ab_save()

Examples

```
# Starting with tabular data
od = leeds_od
od[[1]] = c("E02006876")
zones = leeds_zones
ablines = ab_scenario(od, zones = zones)
ab_list = ab_json(ablines, mode_column = "mode", scenario_name = "test")
ab_list$scenario
f = tempfile(fileext = ".json")
ab_save(ab_list, f)
readLines(f)[1:30]

# Starting with JSON data from A/B Street (multiple trips per person)
f = system.file("extdata/minimal_scenario2.json", package = "abstr")
desire_lines = ab_sf(f)
desire_lines
json_list = ab_json(desire_lines)
json_list
```

---

ab_save *Save OD data as JSON files for import into A/B Street*

Description

Save OD data as JSON files for import into A/B Street

Usage

```
ab_save(x, f)
```

Arguments

x	A list object produced by ab_scenario()
f	A filename, e.g. new_scenario.json

Value

A JSON file containing scenarios from ab_scenario()

---

ab_scenario *Generate A/B Street Scenario objects by disaggregating aggregate OD data*

Description

This function takes a data frame representing origin-destination trip data in 'long' form, zones and, optionally, buildings from where trips can start and end as inputs.

Usage

```
ab_scenario(
  od,
  zones,
  zones_d = NULL,
  origin_buildings = NULL,
  destination_buildings = NULL,
  pop_var = 3,
  time_fun = ab_time_normal,
  output = "sf",
  modes = c("Walk", "Bike", "Transit", "Drive"),
  ...
)
```

Arguments

od	Origin destination data with the first 2 columns containing zone code of origin and zone code of destination. Subsequent columns should be mode names such as All and Walk, Bike, Transit, Drive, representing the number of trips made by each mode of transport for use in A/B Street.
zones	Zones with IDs that match the desire lines. Class: sf.
zones_d	Optional destination zones with IDs that match the second column of the od data frame (work in progress)
origin_buildings	Polygons where trips will originate (sf object)
destination_buildings	Polygons where trips can end, represented as sf object

pop_var	The variable containing the total population of each desire line.
time_fun	The function used to calculate departure times. ab_time_normal() by default.
output	Which output format? "sf" (default) and "json_list" return R objects. A file name such as "baseline.json" will save the resulting scenario to a file.
modes	Character string containing the names of the modes of travel to include in the outputs. These must match column names in the od input data frame. The default is c("Walk", "Bike", "Drive", "Transit"), matching the mode names allowed in the A/B Street scenario schema.
...	Additional arguments to pass to ab_json()

Value

An sf object by default representing individual trips between randomly selected points (or buildings when available) between the zones represented in the OD data.

Examples

```
od = leeds_od
zones = leeds_zones
od[[1]] = c("E02006876")
ablines = ab_scenario(od, zones = zones)
plot(ablines)
table(ablines$mode)
colSums(od[3:7]) # 0.17 vs 0.05 for ab_scenario
ablines = ab_scenario(od, zones = zones, origin_buildings = leeds_buildings)
plot(leeds_zones$geometry)
plot(leeds_buildings$geometry, add = TRUE)
plot(ablines["mode"], add = TRUE)
ablines_json = ab_json(ablines, scenario_name = "test")
od = leeds_desire_lines
names(od)[4:6] = c("Walk", "Bike", "Drive")
ablines = ab_scenario(
  od = od,
  zones = leeds_site_area,
  zones_d = leeds_zones,
  origin_buildings = leeds_houses,
  destination_buildings = leeds_buildings,
  output = "sf"
)
plot(ablines)
plot(ablines$geometry)
plot(leeds_site_area$geometry, add = TRUE)
plot(leeds_buildings$geometry, add = TRUE)
```

---

ab_sf *Convert JSON representation of trips from A/B Street into an 'sf' object*

Description

This function takes a path to a JSON file representing an A/B Street scenario, or an R representation of the JSON in a list, and returns an `sf` object with the same structure as objects returned by `ab_scenario()`.

Usage

```
ab_sf(json)
```

Arguments

`json`	Character string or list representing a JSON file or list that has been read into R and converted to a data frame.

Details

Note: the departure time in seconds is divided by 10000 on conversion to represent seconds, which are easier to work with that 10,000th of a second units.

Value

An `sf` data frame representing travel behaviour scenarios from, and which can be fed into, A/B Street. Contains the following columns: person (the ID of each agent in the simulation), departure (seconds after midnight of the travel starting), mode (the mode of transport, being `Walk`, `Bike`, `Transit` and `Drive`), purpose (what the trip was for, e.g. `Work`), and geometry (a linestring showing the start and end point of the trip/stage).

Examples

```
file_name = system.file("extdata/minimal_scenario2.json", package = "abstr")
ab_sf(file_name)
json = jsonlite::read_json(file_name, simplifyVector = TRUE)
ab_sf(json)
```

---

ab_time_normal *Generate times for A/B scenarios*

Description

Generate times for A/B scenarios

Usage

```
ab_time_normal(hr = 8.5, sd = 0.5, n = 1)
```

Arguments

hr	Number representing the hour of day of departure (on average). 8.5, for example represents 08:30.
sd	The standard deviation in hours of the distribution
n	The number of numbers representing times to return

Value

An integer representing the time since midnight in seconds

Examples

```
time_lunch = ab_time_normal(hr = 12.5, sd = 0.25)
time_lunch
# Back to a formal time class
as.POSIXct(trunc(Sys.time(), units="days") + time_lunch)
time_morning = ab_time_normal(hr = 8.5, sd = 0.5)
as.POSIXct(trunc(Sys.time(), units="days") + time_morning)
time_afternoon = ab_time_normal(hr = 17, sd = 0.75)
as.POSIXct(trunc(Sys.time(), units="days") + time_afternoon)
```

---

leeds_site_area *Datasets from Leeds*

Description

These datasets represent data from the case study city of Leeds.

---

montlake_buildings *Example OSM Buildings Table for Montlake*

Description

Each row of this table contains a building that exists within a zone in the `montlake_zones` table.

Format

A sf dataframe with columns:

osm_way_id OSM ID assigned to each building.

name OSM name assigned to each building (might be NA).

building OSM building category assigned to each building.

geometry Simple feature collection (sfc) contain multipolygons, each representing the boundaries of a building.

Details

These buildings were retrieved using `osmextract::oe_read()`. See the code used to create this data in `data-raw/montlake-test-data.R`.

Source

OpenStreetMap

Examples

```
library(sf)
names(montlake_buildings)
head(montlake_buildings$osm_way_id)
head(montlake_buildings$name)
head(montlake_buildings$building)
nrow(montlake_buildings)
plot(montlake_buildings)
```

---

montlake_od *Example OD Table for Montlake*

Description

Each row of this table contains a count of how many trips started in column `o_id` and ended in column `d_id` according to different modes. This example table has modes that match what A/B Street currently uses: "Drive", "Transit", "Bike", and "Walk".

Usage

```
montlake_od
```

Format

A data frame with columns:

o_id Trip origin zone ID (must match an ID in `montlake_zone`).

d_id Trip destination zone ID (must match an ID in `montlake_zone`).

Drive Count of how many trips were made using cars.

Transit Count of how many trips were made using public transit.

Bike Count of how many trips were made using bikes.

Walk Count of how many trips were made on foot.

Details

See the code used to create this data in "data-raw/montlake-test-data.R"

montlake_zones *Example Zones Table for Montlake*

Description

Each row of this table contains the zone ID and geometry for a TAZ near Montlake. Includes all zones that start or end within the Montlake boundary and have at least 25 trips across all modes of transit in `montlake_od`.

Usage

```
montlake_zones
```

Format

A sf dataframe with columns:

id Zone ID (must match `o_id` or `d_id` in `montlake_od`).

geometry Simple feature collection (sfc) contain multipolygons, each representing the boundaries of a TAZ near Montlake.

Details

See the code used to create this data in "data-raw/montlake-test-data.R"

sao_paulo_activity_df_20 *Example Activity data for São Paulo*

Description

Each row of this table contains a single trip of people in the São Paulo city. `sao_paulo_activity_df_2` represents the movement of 2 people, `sao_paulo_activity_df_20` represents the movement of 20 people.

Usage

```
sao_paulo_activity_df_20
```

Format

A data frame with columns:

ID_PESS Person identifier.

CO_O_X Origin coordinate X.

CO_O_Y Origin coordinate Y.

CO_D_X Destination coordinate X.

CO_D_Y Destination coordinate Y.

MODOPRIN Main mode.

H_SAIDA Departure hour.

MIN_SAIDA Departure minute.

Details

See the code used to create this data, and the full open dataset with 128 variables, the file data-raw/sao-paulo-activity-data.R in the package's GitHub repo.

Examples

```
dim(sao_paulo_activity_df_20)
names(sao_paulo_activity_df_20)
head(sao_paulo_activity_df_20)
dim(sao_paulo_activity_df_2)
names(sao_paulo_activity_df_2)
head(sao_paulo_activity_df_2)
```

---

sao_paulo_activity_sf_20

Example Activity data for São Paulo

Description

Each row of this table contains a single trip of people in the São Paulo city. `sao_paulo_activity_sf_2` represents the movement of 2 people, `sao_paulo_activity_sf_20` represents the movement of 20 people.

Usage

```
sao_paulo_activity_sf_20
```

Format

A data frame with columns:

person Person identifier.

departure Departure time in seconds past midnight

mode Mode of travel in A/B Street terms

purpose Purpose of travel in A/B Street terms

geometry Geometry of the linestring representing the OD pair

Details

See the code used to create this data, and the full open dataset with 128 variables, the file data-raw/sao-paulo-activity-data.R in the package's GitHub repo.

Examples

```
dim(sao_paulo_activity_sf_20)
names(sao_paulo_activity_sf_20)
head(sao_paulo_activity_sf_20)
table(sao_paulo_activity_sf_20$mode)
table(sao_paulo_activity_sf_20$purpose)
dim(sao_paulo_activity_sf_2)
names(sao_paulo_activity_sf_2)
head(sao_paulo_activity_sf_2)
```

Index

∗ **datasets**
    montlake_od, 8
    montlake_zones, 9
    sao_paulo_activity_df_20, 9
    sao_paulo_activity_sf_20, 10

ab_json, 2
ab_json(), *3*, *5*
ab_save, 3
ab_save(), *3*
ab_scenario, 4
ab_scenario(), *2*, *4*, *6*
ab_sf, 5
ab_time_normal, 6

leeds_buildings *(*leeds_site_area*)*, 7
leeds_desire_lines *(*leeds_site_area*)*, 7
leeds_houses *(*leeds_site_area*)*, 7
leeds_od *(*leeds_site_area*)*, 7
leeds_site_area, 7
leeds_zones *(*leeds_site_area*)*, 7

montlake_buildings, 7
montlake_od, 8
montlake_zones, 9

sao_paulo_activity_df_2
        *(*sao_paulo_activity_df_20*)*, 9
sao_paulo_activity_df_20, 9
sao_paulo_activity_sf_2
        *(*sao_paulo_activity_sf_20*)*, 10
sao_paulo_activity_sf_20, 10