Package: ssrn (via r-universe)

November 14, 2026

Title Scan Statistics for Railway Network

Version 0.1.0

Date 2020-06-14

Description Implement the algorithm provided in scan for estimating the transmission route on railway network using passenger volume. It is a generalization of the scan statistic approach for railway network to identify the hot railway route for transmitting infectious diseases.

URL https://github.com/uribo/ssrn

BugReports https://github.com/uribo/ssrn/issues

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.0

Depends R (>= 3.2.0)

Imports dplyr (>= 1.0.0), magrittr (>= 1.5), purrr (>= 0.3.4), rlang (>= 0.4.6), stringr (>= 1.4.0), tibble (>= 3.0.1), tidyr (>= 1.1.0)

Suggests testthat, scanstatistics

Config/pak/sysreqs libicu-dev

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

Date/Publication 2020-06-14 05:23:05 UTC

RemoteUrl https://github.com/uribo/ssrn

RemoteRef HEAD

RemoteSha c0c2837eb6c3d7ff81ae23563b6932b95598fc15

Contents

jreast_jt 2
jreast_jt_od 2
make_adjacency_matrix 3
make_passenger_matrix 3
make_passenger_od 4
network_window 5
transit_table 5

Index **7**

`jreast_jt` *East Japan Railway's Tokaido Line Data*

Description

East Japan Railway's Tokaido Line Data

Details

Includes the names of stations between Tokyo and Yugawara as of June 2020.

- `st_code`: A unique number to identify the station.
- `st_name`: Romanization of station names.

Value

- jreast_jt a tibble

`jreast_jt_od` *JR-East Tokaido Line OD Data*

Description

JR-East Tokaido Line OD Data

Details

Census values made in 2015. The number of passengers between stations on the Tokaido Line. These values are those of commuter pass users.

- `departure_st_code`: Departing station identification number.
- `arrive_st_code`: The identification number of the station you are arriving at.
- `volume` Number of people getting on and off the train.

Value

- jreast_jt_od a tibble

See Also

https://www.mlit.go.jp/sogoseisaku/transport/sosei_transport_tk_000035.html

---

make_adjacency_matrix *Convert station data to adjacency matrix*

Description

Convert station data to adjacency matrix

Usage

```
make_adjacency_matrix(stations, depart, arrive)
```

Arguments

stations	data.frame which set of stopping points recorded in order of stopping.
depart	Column name of a stop.
arrive	Give the name of the column indicating the next stop at the target stop.

Examples

```
make_adjacency_matrix(jreast_jt, st_code, next_st_code)
```

---

make_passenger_matrix *Convert passenger and station data to origin-destination matrix*

Description

Convert passenger and station data to origin-destination matrix

Usage

```
make_passenger_matrix(passenger, stations, depart, arrive, location, value)
```

Arguments

passenger	passenger data
stations	data.frame which set of stopping points recorded in order of stopping.
depart	Column name of a stop.
arrive	Give the name of the column indicating the next stop at the target stop.
location	Name of the variable to use for the join, indicating its location.
value	origin-destination value name

Examples

```
jreast_jt_od %>%
  make_passenger_matrix(jreast_jt,
                        departure_st_code,
                        arrive_st_code,
                        st_code,
                        volume)
```

`make_passenger_od` *Summaries a passenger volume*

Description

Summaries a passenger volume

Usage

```
make_passenger_od(
  passenger,
  stations,
  depart,
  arrive,
  location,
  value,
  .all = FALSE
)
```

Arguments

`passenger`	passenger data
`stations`	data.frame which set of stopping points recorded in order of stopping.
`depart`	Column name of a stop.
`arrive`	Give the name of the column indicating the next stop at the target stop.
`location`	Name of the variable to use for the join, indicating its location.
`value`	origin-destination value name
`.all`	Make a join that contains rows of two datasets. The default value is *FALSE*.

Examples

```
jreast_jt_od %>%
  make_passenger_od(jreast_jt,
                    depart = departure_st_code,
                    arrive_st_code,
                    location = st_code,
                    value = volume) %>%
 dplyr::left_join(jreast_jt %>%
```

```
                    dplyr::select(arrive_st_code = st_code,
                                  next_st_name = st_name),
                  by = "arrive_st_code")
```

`network_window` *Create network window zones*

Description

Create network window zones

Usage

```
network_window(adjacency_matrix, dist_matrix, type, cluster_max)
```

Arguments

`adjacency_matrix`	A boolean matrix, with element (*i*,*j*) set to TRUE if location *j* is adjacent to location *i*.
`dist_matrix`	Distance matrix
`type`	Currently, "connected_B" only.
`cluster_max`	Maximum cluster size. Zone If this value is reached, the area will not be expanded any further. It's a good idea to keep it to the number of stops on the line you're dealing with.

`transit_table` *Create transit table*

Description

Create transit table

Usage

```
transit_table(stations, ..., reverse = FALSE)
```

Arguments

`stations`	data.frame which set of stopping points recorded in order of stopping.
`...`	Arguments passed on to `dplyr::across` `.cols <tidy-select>` Columns to transform. Because `across()` is used within functions like `summarise()` and `mutate()`, you can't select or compute upon grouping variables.
`reverse`	Option to swap the order of the stopping points.

Examples

```
# The next stop is stored in the variable of column next_.
jreast_jt %>%
  transit_table()
# Switch between inbound and outbound lines.
jreast_jt %>%
  transit_table(reverse = TRUE)
```

Index

dplyr::across, 5

jreast_jt, 2
jreast_jt_od, 2

make_adjacency_matrix, 3
make_passenger_matrix, 3
make_passenger_od, 4

network_window, 5

transit_table, 5