Package 'movr'

December 17, 2015

Type Package

Version 0.1.3

Title Analyzing and Visualizing Human Mobility Data

Description A set of tools to analyze and visualize spatio-temporal mobility data.

Author Xiaming Chen <chen@xiaming.me>

Maintainer Xiaming Chen <chen@xiaming.me>

URL https://github.com/caesar0301/movr

BugReports https://github.com/caesar0301/movr/issues

Depends R (>= 3.0.0), dplyr, tidyr, data.table, geosphere

Suggests ggplot2, knitr

LazyData yes

SystemRequirements GNU CMake

VignetteBuilder knitr

R topics documented:

`cart2geo` *Cartesian and geopoint conversion*

Description

Converting Cartesian coordinates into long/lat geo-points.

Usage

```
cart2geo(x)
```

Arguments

x	A unit-length 3D vector (x, y, z) in Cartesian system.

Value

A 2D vector (lat, long) representing the geo-point in degree.

See Also

`geo2cart`, `cart2geo.radian`

Examples

```
cart2geo(c(-0.4330127, 0.7500000, 0.5000000))
```

cart2geo.radian *Convert Cartesian coordinates to geopoints in radians.*

Description

Convert Cartesian coordinates to geopoints in radians.

Usage

```
cart2geo.radian(x)
```

Arguments

x	A 2D vector (lat, long) representing the geo-point in radians.

See Also

geo2cart.radian

compress_mov *Compress movement history*

Description

Remove duplicate location records in user's movement history. Continuous records at the same location is merged into a single session (with interval less than 'gap') recording the starting and ending times.

Usage

```
compress_mov(x, y = NULL, t = NULL, gap = 0.5 * 3600)
```

Arguments

x,y,t	see params of stcoords
gap	the time tolerance (sec) to conbime two continuous observations

See Also

flowmap, flowmap2, flow_stat, draw_flowmap, stcoords

Examples

```
data(movement)

user_move <- subset(movement, id==1)
compress_mov(user_move[,c("loc", "time")])

## With dplyr
library(dplyr)
movement %>% dplyr::filter(id<10) %>%
 group_by(id) %>% do(compress_mov(x=.$loc, t=.$time))
```

deg2rad *Convert degrees to radians.*

Description

Convert degrees to radians.

Usage

```
deg2rad(deg)
```

Arguments

deg	A number or vector of degrees.

See Also

rad2deg

draw_flowmap *Visualize flowmap.*

Description

Visualize the mobility statistics (flowmap) from data. Each row in the data will generate a line on the map.

Usage

```
draw_flowmap(from_lat, from_lon, to_lat, to_lon, dist.log = TRUE,
  weight = NULL, weight.log = TRUE, gc.breaks = 5, col.pal = c("white",
  "blue", "black"), col.pal.bias = 0.3, col.pal.grad = 200,
  new.device = TRUE, bg = "black", ...)
```

Arguments

from_lat	The latitude coordinates of departing point for mobile transitions.
from_lon	The longitude coordinates of departing point for mobile transitions.
to_lat	The latitude coordinates of arriving point for mobile transitions.
to_lon	The longitude coordinates of arriving point for mobile transitions.
dist.log	Whether using log-scale distance for line color.
weight	The user-defined weight for line color. Larger weight corresponds to lefter color of col.pal.
weight.log	Whether using log-scale weight for line color.
gc.breaks	The number of intermediate points (excluding two ends) to draw a great circle path.
col.pal	A color vector used by colorRampPalette; must be a valid argument to col2rgb. Refer to colorbrewer2.org to derive more palettes.

col.pal.bias	The bias coefficient used by colorRampPalette. Higher values give more widely spaced colors at the high end.
col.pal.grad	The number of color grades to diffeciate distance.
new.device	Whether creating a new device for current plot. Set this parameter as FALSE when trying to plot multiple flowmaps in one figure.
bg	The background color for current plot. It is working when new.device is TRUE.
...	Extra parameters for basic plot() function.

See Also

compress_mov, flowmap, flowmap2, flow_stat

---

fit_polyexp *Fit a poly-exponential distribution*

Description

Model: y ~ exp(a*x^2 + b*x + c) * x^d

Usage

```
fit_polyexp(x, y, xmin = min(x), xmax = max(x), plot = TRUE, add = TRUE,
  ...)
```

Arguments

x	A vector of independent variable.
y	A vector of dependent variable.
xmin	The lower bound point of x.
xmax	The higher truncated point of x.
plot	Whether to plot the fitted curve.
add	Whether to add the fitted curve to current plot.
...	Extra parameters to curve.

Value

A list of values for a, b, c, and d.

`fit_power_law` *Fit a power-law*

Description

Model: y ~ a * x^-lambda

Usage

```
fit_power_law(x, y, xmin = min(x), xmax = max(x), plot = TRUE,
  add = TRUE, ...)
```

Arguments

`x`	A vector of independent variable.
`y`	A vector of dependent variable.
`xmin`	The lower bound point of x.
`xmax`	The higher truncated point of x.
`plot`	Whether to plot the fitted curve.
`add`	Whether to add the fitted curve to current plot.
`...`	Extra parameters to `curve`.

Value

A list of values for a and lambda.

See Also

`fit_power_law`, `fit_truncated_power_law`

`fit_truncated_power_law` *Fit a truncated power-law.*

Description

Model: y ~ a * x^(-lambda) exp(-x/k)

Usage

```
fit_truncated_power_law(x, y, xmin = min(x), xmax = max(x), plot = TRUE,
  add = TRUE, ...)
```

Arguments

x	A vector of independent variable.
y	A vector of dependent variable.
xmin	The lower bound point of x.
xmax	The higher truncated point of x.
plot	Whether to plot the fitted curve.
add	Whether to add the fitted curve to current plot.
...	Extra parameters to curve.

Value

A list of values for a, lambda and k.

See Also

fit_power_law

---

flowmap *Generate flowmap from movement data*

Description

Use historical movement data to generate flowmap, which records mobility statistics between two locations 'from' and 'to'.

Usage

```
flowmap(uid, loc, time, gap = 8 * 3600)
```

Arguments

uid	a vector to record user identities
loc	a 1D vector to record locations of movement history
time	the timestamp (SECONDS) vector of movement history
gap	the maximum dwelling time to consider a valid move between locations.

Value

a data frame with four columns: from, to, total, unique (users)

See Also

compress_mov, flowmap2, flow_stat, draw_flowmap

Examples

```
data(movement)

with(movement, flowmap(id, loc, time))
```

flowmap2 *Generate flowmap from movement data*

Description

Use historical movement data to generate flowmap, which records mobility statistics between two locations 'from' and 'to'.

Usage

```
flowmap2(uid, loc, stime, etime, gap = 86400)
```

Arguments

uid	a vector to record user identities
loc	a 1D vector to record locations of movement history
stime,etime	compressed session time at each location
gap	the maximum dwelling time to consider a valid move between locations

Details

Different from `flowmap`, compressed movement history is used to generate flow statistics.

Value

a data frame with four columns: from, to, total, unique (users)

See Also

compress_mov, flowmap, flow_stat, draw_flowmap

---

flow_stat *Calculate flow stat between locations*

Description

Calculate flow stat between locations

Usage

```
flow_stat(loc, stime, etime, gap = 86400)
```

Arguments

loc	A 1D vector to record locations of movement history
stime	The starting timestamp (SECONDS) vector of movement history
etime	The ending timestamp (SECONDS) vector of movement history
gap	The temporal idle interval

See Also

compress_mov, flowmap, flowmap2, draw_flowmap

Examples

```
data(movement)

user_move <- subset(movement, id==1)
sessions <- compress_mov(user_move[,c("loc", "time")])

with(sessions, flow_stat(loc, stime, etime))
```

---

gcd *Great Circle Distance (GCD)*

Description

Calculates the geodesic distance between two points specified by radian latitude/longitude using one of the Spherical Law of Cosines (slc), the Haversine formula (hf), or the Vincenty inverse formula for ellipsoids (vif).

Usage

```
gcd(p1, p2, type = "slc")
```

Arguments

p1	Location of point 1 with (lat, long) coordinates.
p2	Location of point 2 with (lat, long) coordinates.
type	Specific algorithm to use, c('slc', 'hf', 'vif').

Value

Distance in kilometers (km).

References

http://www.r-bloggers.com/great-circle-distance-calculations-in-r/

Examples

```
# Point in (lat, long) format
p1 <- c(30.0, 120.0)
p2 <- c(30.5, 120.5)

gcd(p1, p2)
gcd(p1, p2, type="hf")
gcd(p1, p2, type="vif")
```

geo2cart — *Geopoint and Cartesian conversion*

Description

Converting geo-points in lat/long into Cartesian coordinates.

Usage

```
geo2cart(x)
```

Arguments

x	A 2D vector (lat, long) representing the geo-point in degrees.

Value

A unit-length 3D vector (x, y, z) in Cartesian system.

See Also

geo2cart.radian, cart2geo

Examples

```
geo2cart(c(30, 120))
```

geo2cart.radian — *Convert geopoints in radians to Cartesian coordinates.*

Description

Convert geopoints in radians to Cartesian coordinates.

Usage

```
geo2cart.radian(x)
```

Arguments

x	A 2D vector (lat, long) representing the geo-point in radians.

See Also

cart2geo.radian

in_area *Geographic area checking*

Description

Check if the given lon-lat pair falls into specific area. The area is a 4-length vector with lon-lat pairs of two points that confine the area boundaries.

Usage

```
in_area(lon, lat, area)
```

Arguments

lon,lat	The point to be checked.
area	The area defined by two points c(lon1, lat1, lon2, lat2).

Examples

```
in_area(120.1, 30.1, c(120.0,30.0,120.5,30.5))
```

map3d *Add a 3D map surface*

Description

This method add a 3D map surface to the RGL plot. The backend map service is supported by OpenStreetMap package. All parameters except for h are consistent with the 'openmap' function in OSM.

Usage

```
map3d(upperLeft, lowerRight, h = 0, ...)
```

Arguments

upperLeft	the upper left lat and long
lowerRight	the lower right lat and long
h	the horizontal plane to locate the map surface
...	all other parameters of openmap

See Also

openmap

Examples

```
data(movement)
u1 <- subset(movement, id==3)
u1$time <- (u1$time - min(u1$time)) / 3600
region.lat1 <- min(u1$lat) - 0.005
region.lat2 <- max(u1$lat) + 0.005
region.lon1 <- min(u1$lon) - 0.005
region.lon2 <- max(u1$lon) + 0.005

## Not run:
rgl.clear()
rgl.clear("lights")
rgl.bg(color="lightgray")
rgl.viewpoint(theta=30, phi=45)
rgl.light(theta = 45, phi = 45, viewpoint.rel=TRUE)
map3d(c(region.lat2, region.lon1), c(region.lat1, region.lon2),
      min(u1$time), 10, "esri")

axes3d(edges = "bbox", labels = TRUE, tick = TRUE, nticks = 5, box=FALSE,
       expand = 1.03, col="black", lwd=0.8)

## End(Not run)
```

`match_as` *Approximately match*

Description

Match x to y approximately, and return the index of y, which is mostly near to each value in x. A variate of match() or

Usage

```
match_as(x, y)
```

Arguments

x	A given vector to be matched
y	A target vector to calculate absolute approximation

See Also

seq_along, rep_each

Examples

```
a <- c(1,2,3)
b <- c(0.1, 0.2, 0.5)
match_as(a, b)
```

melt_time *Melt time into parts*

Description

Melt time into parts

Usage

```
melt_time(epoch, tz = "Asia/Shanghai")
```

Arguments

epoch	the UNIX epoch timestamp in seconds
tz	the time zone string

Value

several fields (indexed by order) of given timestamp: year, month, day, hour, minute, second, day of week (dow), day of year (doy), week of month (wom), week of year (woy), quarter of year (qoy)

midpoint *Geographic midpoint calculation*

Description

Calculate the midpoint given a list of locations denoted by latitude and longitude coordinates.

Usage

```
midpoint(lat, lon, w = rep(1, length(lat)))
```

Arguments

lat,lon	The location points
w	The weighted value for each point

Value

The geographic midpoint in lat/lon

References

http://www.geomidpoint.com/calculation.html

See Also

radius_gyration

Examples

```
lat <- c(30.2, 30, 30.5)
lon <- c(120, 120.4, 120.5)

# equal weight
midpoint(lat, lon)

# custom weight
w <- c(1, 2, 1)
midpoint(lat, lon, w)
```

---

movr *movr: inspecting human mobility with R*

Description

A package targeting at analyzing, modeling, and visualizing human mobility from temporal and spatial perspectives.

---

rad2deg *Convert radians to degrees.*

Description

Convert radians to degrees.

Usage

```
rad2deg(rad)
```

Arguments

rad	A number or vector of radians

See Also

deg2rad

radius_gyration *Radius of gyration for human mobility*

Description

Given a series of locations denoted by lat/lon coordinates, the radius of gyration for individual is calculated.

Usage

```
radius_gyration(lat, lon, w = rep(1, length(lat)))
```

Arguments

lat,lon	The geographic coordinates of locations
w	The weight value for each location

Value

The radius of gyration (km)

References

M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," Nature, vol. 453, no. 7196, pp. 779-782, Jun. 2008.

See Also

midpoint

Examples

```
lat <- c(30.2, 30, 30.5)
lon <- c(120, 120.4, 120.5)
radius_gyration(lat, lon)
```

Rcolors *R Colors*

Description

Plot matrix of R colors, in index order, 25 per row. This is for quick reference when programming.

Usage

```
Rcolors(huesort=TRUE)
```

Arguments

huesort	Boolean value to control ordering by HUE.

Details

Copyright: Earl F. Glynn

References

http://research.stowers-institute.org/efg/R/Color/Chart/

`rep_each` *Replicate elements of vector*

Description

This is a slight modification of `rep` in basic package. It replicates each element of a vector one by one to construct a new vector.

Usage

```
rep_each(x, times = 2)
```

Arguments

`x`	a vector
`times`	the number of replication times of each element.

See Also

`match_as`,

Examples

```
rep(1:10, 2)
rep_each(1:10, 2)
```

`RMSE` *Root Mean Squared Error (RMSE)*

Description

calculate the root mean squared error (RMSE) of two vectors.

Usage

```
RMSE(x, y)
```

Arguments

`x`	A given vector to calculate RMSE.
`y`	The target vector

Examples

```
RMSE(c(1,2,3,4), c(2,3,2,3))
```

seq_collapsed — *Sequencing by collapsing adjacent same values*

Description

Generate integer sequence by assigning the same adjacent values to the same level.

Usage

```
seq_collapsed(v)
```

Arguments

v	The input vector.

See Also

seq_along, seq_distinct, vbin, vbin_range, vbin_grid

Examples

```
seq_collapsed(c(1,2,2,3,2,2))
```

seq_distinct — *Sequencing by distinct values*

Description

Generate a new (integer) sequence according to distinct value levels. The same value takes a unique order number.

Usage

```
seq_distinct(v)
```

Arguments

v	A vector to generate integer sequence

See Also

seq_along, seq_collapsed, vbin, vbin_range, vbin_grid

Examples

```
seq_along(c(1,2,3,2))
seq_distinct(c(1,2,3,2))

## See also
library(tidyr)
seq_range(c(1,2,3,2), 3)
```

standardize — *Vector Normalization*

Description

Normalize a given vector.

Usage

```
standardize(x)
```

Arguments

x	A vector to be normalized.

See Also

standardize_st

Examples

```
standardize(c(1,2,3,4,5,6))
```

standardize_st — *Normalization over spatial and temporal scale*

Description

Scale the value along spatial and temporal coordinates simultaneously.

Usage

```
standardize_st(scoord, tcoord, value, alpha = 0.5)
```

Arguments

scoord	a 1D vector of spatial coordinate
tcoord	a 1D vector of temporal coordinate
value	a value vector for each (scoord, tcoord)
alpha	a tuning parameter controling the weight of space and time

Examples

```
scoord <- rep(seq(6), 2)
tcoord <- rep(c(1,2), each=6)
value <- runif(6 * 2)
standardize_st(scoord, tcoord, log10(1+value))
```

`stcoords` *Spatiotemporal data formatting*

Description

Format spatiotemporal series in a unified manner for both 1D and 2D locations. If `x` is a data frame or matrix, `y` and `t` are omitted.

Usage

```
stcoords(x, y = NULL, t = NULL)
```

Arguments

`x`	A vector, data frame or matrix.
`y`	A vector.
`t`	A vector.

Details

If `x` is a data frame (3 columns), this function automatically identify spatial and temporal values by column names, i.e., (x,y,t) and (lat,lon,time). Otherwise, the column indexes are employed as [, 1] and [, 2] being the space coordinates and [, 3] being the timestamps.

If `x` is a data frame (2 columns), similar policies are involved, but alternatively column names (x, t) and (loc, time) are used.

If `x` is a matrix, column indexes are used merely.

If `x` is a vector, dimensions of space coordinates are determined by both `x` and `y`, and the time dimension by `t`.

See Also

`stcoords_1d`

Examples

```
## One data frame with columes x, y, t
x <- data.frame(x=rep(1:10, 2), y=rep_each(1:10, 2), t=1:20)
stcoords(x)

## One data frame without demanded colume names
x <- data.frame(rep(1:10, 2), rep_each(1:10, 2), 1:20)

## One data frame with two columes loc, time
x <- data.frame(loc=rep(1:10, 2), time=1:20)

## With vectors
stcoords(x=rep(1:10, 2), t=1:20)
```

`stcoords_1d` *Spatiotemporal data formatting (1D)*

Description

Similar to `stcoords`, return location instead of x, y coordinates.

Usage

```
stcoords_1d(x, y = NULL, t = NULL)
```

Arguments

`x,y,t`	params of `stcoords`

See Also

`stcoords`

Examples

```
x <- data.frame(rep(1:10, 2), rep_each(1:10, 2), 1:20)
stcoords_1d(x)
```

`vbin` *Vector binning*

Description

Bin a vector into ‘n‘ intervals in regard with its value range. The vector x is split into n bins within [min(x), max(x)], and bin index is given by checking the bin [bin_min, bin_max) into which data points in x fall.

Usage

```
vbin(x, n, center=c(TRUE, FALSE))
```

Arguments

`x`	a numeric vector
`n`	the number of bins
`center`	indication of representing intervals as index (default) or center points.

Value

Sequence with interval index or center points.

See Also

`match_as`, `vbin_range`, `vbin_grid`

Examples

```
vbin(1:10, 3)
vbin(1:10, 3, TRUE)
```

---

vbin_grid — *2D random field binning*

Description

Generate a bined matrix given a 2D random field.

Usage

```
vbin_grid(x, y, z, nx, ny, FUN = mean, na = NA)
```

Arguments

x,y,z	a random field with location vectors (x, y) and value vector z. They must have the same length.
nx,ny	the number of bins in x and y dimension.
FUN	a function to calculate statistics in each 2D bin.
na	Replacemnet for NA value in matrix bins.

Value

a matrix with row (column) names being the center points of x (y) dim, and with cell value being the aggregate statistics calculated by FUN.

See Also

match_as, vbin, vbin_range

Examples

```
vbin_grid(1:20, 20:1, runif(20), nx=5, ny=5)
```

---

vbin_range — *Vector range binning*

Description

Bin the range of given vector into n itervals.

Usage

```
vbin_range(x, n)
```

Arguments

x	a numeric vector
n	the number of bins

Value

the center of each interval

See Also

match_as, vbin, vbin_grid

Examples

```
vbin_range(10:20, 3)
```

Index