Package: TDLM (via r-universe)

October 13, 2024

Type Package

Title Systematic Comparison of Trip Distribution Laws and Models

Version 1.0.0

Description The main purpose of this package is to propose a rigorous framework to fairly compare trip distribution laws and models as described in Lenormand et al. (2016) <doi:10.1016/j.jtrangeo.2015.12.008>.

Depends R (>= 4.0.0)

License GPL-3

Encoding UTF-8

LazyData true

LazyDataCompression xz

Imports Ecume, mathjaxr, Rdpack(>= 1.0.0), readr (>= 2.0.0), rmarkdown (>= 2.0.0), sf (>= 1.0.0)

RdMacros mathjaxr, Rdpack

Suggests knitr, testthat (>= 3.0.0)

VignetteBuilder knitr

RoxygenNote 7.2.3

BugReports https://github.com/EpiVec/TDLM/issues

Roxygen list(markdown = TRUE)

URL https://epivec.github.io/TDLM/

Config/testthat/edition 3

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

RemoteUrl https://github.com/epivec/tdlm

RemoteRef HEAD

RemoteSha b84130196c781435b42bc32f6832fdf40847a04b

Contents

`calib_param` *Automatic calibration of trip distribution laws' parameter*

Description

This function returns an estimation of the optimal parameter value based on the average surface area of the locations (in square kilometer) according to the law. This estimation has only been tested on commuting data (in kilometer).

Usage

```
calib_param(av_surf, law = "NGravExp")
```

Arguments

`av_surf`	a positive numeric value indicating the average surface area of the locations (in square kilometer).
`law`	a character indicating which law to use (see Details).

Details

The estimation is based on the Figure 8 in Lenormand et al. (2016) for four types of laws. The normalized gravity law with an exponential distance decay function (`law = "NGravExp"`), the normalized gravity law with a power distance decay function (`law = "NGravPow"`), the Schneider's intervening opportunities law (`law = "Schneider"`) and the extended radiation law (`law = "RadExt"`).

Value

An estimation of the optimal parameter value based on the average surface area of the locations.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

References

Lenormand M, Bassolas A, Ramasco JJ (2016). "Systematic comparison of trip distribution laws and models." *Journal of Transport Geography*, **51**, 158-169.

See Also

`extract_opportunities()` `extract_spatial_information()` `check_format_names()`

Examples

```
data(county)

res <- extract_spatial_information(county, id = "ID")
av_surf <- mean(res$surface)

calib_param(av_surf = av_surf, law = "NGravExp")
calib_param(av_surf = av_surf, law = "NGravPow")
calib_param(av_surf = av_surf, law = "Schneider")
calib_param(av_surf = av_surf, law = "RadExt")
```

---

check_format_names *Check format of TDLM's inputs*

Description

This function checks that the TDLM's inputs have the required format (an names).

Usage

```
check_format_names(vectors, matrices = NULL, check = "format_and_names")
```

Arguments

`vectors`	a list of vectors. The list can contain one vector. It is recommended to name each element of the list. If `vectors = NULL` only the matrices will be considered.
`matrices`	a list of matrices. The list can contain one matrix. It is recommended to name each element of the list. If `matrices = NULL` only the vectors will be considered (by default).
`check`	a character indicating what types of check ("format" or "format_and_names") should be used (see Details).

Details

The TDLM's inputs should be based on the same number of locations sorted in the same order. check = "format" will run basic checks to ensure that the structure of the inputs (dimensions, class, type...) is correct.

It is recommended to use the location ID as vector names, matrix rownames and matrix colnames. Set check = "format_and_names" to check the inputs' names. The checks are run successively, so run the function as many times as needed to get the message indicating that the inputs passed the check successfully.

Value

A message indicating if the check has passed or failed.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

Examples

```
data(mass)
data(distance)

mi <- as.numeric(mass[, 1])
names(mi) <- rownames(mass)
mj <- mi

check_format_names(
  vectors = list(mi = mi, mj = mj),
  matrices = list(distance = distance),
  check = "format_and_names"
)
```

---

county *Spatial distribution of US Kansas counties in 2000*

Description

A dataset containing the geometry of 105 US Kansas counties.

Usage

```
county
```

Format

ID County ID.

Longitude Longitude coordinate of the centroid of the county.

Latitude Latitude coordinate of the centroid of the county.

Area Surface area of the county (in square kilometer).

geometry Geometry of the county.

Source

https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.html

---

distance *Great-circle distances between US Kansas counties*

Description

A dataset containing the great-circle distance (in kilometer) between 105 US Kansas counties.

Usage

```
distance
```

Format

A `matrix` with 105 rows and 105 columns. Each element of the matrix represents the distance between two counties. County ID as rownames and colnames.

Source

https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.html

---

extract_opportunities *Compute the number of opportunities between pairs of locations*

Description

This function computes the number of opportunities between pairs of locations as defined in Lenormand et al. (2016). For a given pair of location the number of opportunities between the location of origin and the location of destination is based on the number of opportunities in a circle of radius equal to the distance between origin and destination centered in the origin. The number of opportunities at origin and destination are not included.

Usage

```
extract_opportunities(opportunity, distance, check_names = FALSE)
```

Arguments

opportunity	a numeric vector representing the number of opportunities per location. The value should be positive.
distance	a squared matrix representing the distance between locations.
check_names	a boolean indicating if the ID location are used as vector names, matrix rownames and colnames and if they should be checked (see Note).

Value

A squared matrix in which each element represents the number of opportunities between a pair of locations.

Note

`opportunity` and `distance` should be based on the same number of locations sorted in the same order. It is recommended to use the location ID as vector names, matrix rownames and matrix colnames and to set `check_names = TRUE` to verify that everything is in order before running this function (`check_names = FALSE` by default). Note that the function `check_format_names()` can be used to control the validity of all the inputs before running the main package's functions.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

References

Lenormand M, Bassolas A, Ramasco JJ (2016). "Systematic comparison of trip distribution laws and models." *Journal of Transport Geography*, **51**, 158-169.

See Also

`calib_param()` `extract_spatial_information()` `check_format_names()`

Examples

```
data(mass)
data(distance)

opportunity <- mass[, 1]

sij <- extract_opportunities(
  opportunity = opportunity,
  distance = distance,
  check_names = FALSE
)
```

`extract_spatial_information` *Extract distances and surface areas from a spatial object*

Description

This function returns a matrix of distances between locations (in kilometer) along with a vector surface areas of the locations (in square kilometer).

Usage

```
extract_spatial_information(geometry, id = NULL, show_progress = FALSE)
```

Arguments

`geometry`	a spatial object that can be handled by the `sf` package.
`id`	name or number of the column to use as rownames and colnames for the output distance matrix (optional, NULL by default). A vector with length equal to the number of locations can also be used.
`show_progress`	a boolean indicating if a progress bar should be displayed.

Details

The `geometry` must be projected in a valid coordinate reference system. It will be reprojected in degrees longitude/latitude to compute the great-circle distances between centroids' locations with an internal function and to compute the surface area with the function st_area from the sf package.

Value

A list composed of two elements. The first element is a squared matrix representing the great-circle distance (in kilometer) between locations. The second element is a vector containing the surface area of each location (in square kilometer).

Note

The outputs are based on the locations contained in `geometry` and sorted in the same order. An optional `id` can also be provided to be used as names for the outputs.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

See Also

`calib_param()` `extract_opportunities()` `check_format_names()`

Examples

```
data(county)

res <- extract_spatial_information(county, id = "ID")

dim(res$distance)

length(res$surface)
```

---

gof *Compute goodness-of-fit measures between observed and simulated OD matrices*

Description

This function returns a data.frame where each row provides one or several goodness-of-fit measures between a simulated and an observed Origin-Destination matrix.

Usage

```
gof(
  sim,
  obs,
  measures = "all",
  distance = NULL,
  bin_size = 2,
  use_proba = FALSE,
  check_names = FALSE
)
```

Arguments

sim	an object of class TDLM (output of run_law_model(), run_law() or run_model()). A matrix or a list of matrices can also be used (see Note).
obs	a squared matrix representing the observed mobility flows.
measures	a vector of string(s) indicating which goodness-of-fit measure(s) to chose (see Details). If "all" is specified, then all measures will be calculated.
distance	a squared matrix representing the distance between locations. Only necessary for the distance-based measures.
bin_size	a numeric value indicating the size of bin used to discretize the distance distribution to compute CPC_d (2 "km" by default).
use_proba	a boolean indicating if the proba matrix should be used instead of the simulated OD matrix to compute the measure(s). Only valid for the output from run_law_model() with argument write_proba = TRUE (see Note).
check_names	a boolean indicating if the ID location are used as matrix rownames and colnames and if they should be checked (see Note).

Details

With n the number of locations, T_{ij} the observed flow between location i and location j (argument `obs`), $\tilde{T}_{ij}$ a simulated flow between location i and location j (a matrix from argument `sim`), $N = \sum_{i,j=1}^{n} T_{ij}$ the sum of observed flows and $\tilde{N} = \sum_{i,j=1}^{n} \tilde{T}_{ij}$ the sum of simulated flows.

Several goodness-of-fit measures have been considered `measures = c("CPC", "NRMSE", "KL", "CPL", "CPC_d", "KS")`. The Common Part of Commuters (Gargiulo et al. 2012; Lenormand et al. 2012; Lenormand et al. 2016),

$$CPC(T, \tilde{T}) = \frac{2 \cdot \sum_{i,j=1}^{n} min(T_{ij}, \tilde{T}_{ij})}{N + \tilde{N}}$$

the Normalized Root Mean Square Error (NRMSE),

$$NRMSE(T, \tilde{T}) = \sqrt{\frac{\sum_{i,j=1}^{n} (T_{ij} - \tilde{T}_{ij})^2}{N}}$$

the Kullback–Leibler divergence (Kullback and Leibler 1951),

$$KL(T, \tilde{T}) = \sum_{i,j=1}^{n} \frac{T_{ij}}{N} \log \left(\frac{T_{ij}}{N} \frac{\tilde{N}}{\tilde{T}_{ij}} \right)$$

the Common Part of Links (CPL) (Lenormand et al. 2016),

$$CPL(T, \tilde{T}) = \frac{2 \cdot \sum_{i,j=1}^{n} 1_{T_{ij}>0} \cdot 1_{\tilde{T}_{ij}>0}}{\sum_{i,j=1}^{n} 1_{T_{ij}>0} + \sum_{i,j=1}^{n} 1_{\tilde{T}_{ij}>0}}$$

the Common Part of Commuters based on the disance (Lenormand et al. 2016), noted CPC_d. Let us consider N_k (and $\tilde{N}_k$) the sum of observed (and simulated) flows at a distance comprised in the bin [`bin_size`*k-`bin_size`, `bin_size`*k[.

$$CPC_d(T, \tilde{T}) = \frac{2 \cdot \sum_{k=1}^{\infty} min(N_k, \tilde{N}_k)}{N + \tilde{N}}$$

and the Kolmogorv-Smirnov statistic and p-value (Massey 1951) , noted KS. It is based on the observed and simulated flow distance distribution and computed with the ks_test function from the Ecume package.

Value

A data.frame providing one or several goodness-of-fit measure(s) between simulated OD(s) and an observed OD. Each row corresponds to a matrix sorted according to the list (or list of list) elements (names are used if provided).

Note

By default, if `sim` is an output of `run_law_model()` the measure(s) are computed only for the simulated OD matrices and not the `proba` matrix (included in the output when `write_proba = TRUE`). The argument `use_proba` can be used to compute the measure(s) based on the `proba` matrix instead of the simulated OD matrix. In this case the argument `obs` should also be a proba matrix.

All the inputs should be based on the same number of locations sorted in the same order. It is recommended to use the location ID as matrix rownames and matrix colnames and to set `check_names = TRUE` to verify that everything is in order before running this function (`check_names = FALSE` by default). Note that the function `check_format_names()` can be used to control the validity of all the inputs before running the main package's functions.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

References

Lenormand M, Bassolas A, Ramasco JJ (2016). "Systematic comparison of trip distribution laws and models." *Journal of Transport Geography*, **51**, 158-169.

Gargiulo F, Lenormand M, Huet S, Baqueiro Espinosa O (2012). "Commuting network model: getting to the essentials." *Journal of Artificial Societies and Social Simulation*, **15**(2), 13.

Lenormand M, Huet S, Gargiulo F, Deffuant G (2012). "A Universal Model of Commuting Networks." *PLoS ONE*, **7**, e45985.

Kullback S, Leibler RA (1951). "On Information and Sufficiency." *The Annals of Mathematical Statistics*, **22**(1), 79 – 86.

Massey FJ (1951). "The Kolmogorov-Smirnov test for goodness of fit." *Journal of the American Statistical Association*, **46**(253), 68–78.

See Also

`run_law_model()` `run_law()` `run_model()` `run_law_model()` `check_format_names()`

Examples

```
data(mass)
data(distance)
data(od)

mi <- as.numeric(mass[, 1])
mj <- mi
Oi <- as.numeric(mass[, 2])
Dj <- as.numeric(mass[, 3])

res <- run_law_model(
  law = "GravExp", mass_origin = mi, mass_destination = mj,
  distance = distance, opportunity = NULL, param = 0.01,
  model = "DCM", nb_trips = NULL, out_trips = Oi, in_trips = Dj,
  average = FALSE, nbrep = 1, maxiter = 50, mindiff = 0.01,
  write_proba = FALSE,
  check_names = FALSE
)

gof(
  sim = res, obs = od, measures = "CPC", distance = NULL, bin_size = 2,
  use_proba = FALSE,
  check_names = FALSE
)
```

mass — *Population and number of out- and in-commuters by US Kansas county in 2000*

Description

A dataset containing the number of inhabitants, in-commuters and out-commuters for 105 US Kansas counties in 2000.

Usage

```
mass
```

Format

A `data.frame` with 105 rows and 3 columns:

rownames County ID.

Population Number of inhabitants.

Out-commuters Number of out-commuters.

In-commuters Number of in-commuters.

Source

https://www2.census.gov/programs-surveys/decennial/tables/2000/county-to-county-worker-flow-files/

od — *Origin-Destination commuting matrix between US Kansas counties in 2000*

Description

A dataset containing the number of commuters between 105 US Kansas counties in 2000.

Usage

```
od
```

Format

A `matrix` with 105 rows and 105 columns. Each element of the matrix represents the number of commuters between two counties. County ID as rownames and colnames.

Source

https://www2.census.gov/programs-surveys/decennial/tables/2000/county-to-county-worker-flow-files/

run_law *Estimate mobility flows based on different trip distribution laws*

Description

This function estimates mobility flows using different distribution laws. As described in Lenormand et al. (2016), we propose a two-step approach to generate mobility flows by separating the trip distribution law, gravity or intervening opportunities, from the modeling approach used to generate the flows from this law. This function only uses the first step to generate a probability distribution based on the different laws.

Usage

```
run_law(
  law = "Unif",
  mass_origin,
  mass_destination = mass_origin,
  distance = NULL,
  opportunity = NULL,
  param = NULL,
  check_names = FALSE
)
```

Arguments

law	a character indicating which law to use (see Details).
mass_origin	a numeric vector representing the mass at origin (i.e. demand).
mass_destination	a numeric vector representing the mass at destination (i.e. attractiveness).
distance	a squared matrix representing the distance between locations (see Details).
opportunity	a squared matrix representing the number of opportunities between locations (see Details). Can be easily computed with extract_opportunities().
param	a vector of numeric value(s) used to adjust the importance of distance or opportunity associated with the chosen law. A single value or a vector of several parameter values can be used (see Return). Not necessary for the original radiation law or the uniform law (see Details).
check_names	a boolean indicating if the ID location are used as vector names, matrix rownames and colnames and if they should be checked (see Note).

Details

We compute the matrix proba estimating the probability p_{ij} to observe a trip from location i to another location j ($\sum_i \sum_j p_{ij} = 1$). This probability is based on the demand m_i (argument mass_origin) and the attractiveness m_j (argument mass_destination). Note that the population is typically used as a surrogate for both quantities (this is why mass_destination = mass_origin

by default). It also depends on the distance d_{ij} between locations (argument `distance`) OR the number of opportunities s_{ij} between locations (argument `opportunity`) depending on the chosen law. Both the effect of the distance and the number of opportunities can be adjusted with a parameter (argument `param`) except for the original radiation law or the uniform law.

In this package we consider eight probabilistic laws described in details in Lenormand et al. (2016). Four gravity laws (Carey 1858; Zipf 1946; Barthelemy 2011; Lenormand et al. 2016), three intervening opportunity laws (Schneider 1959; Simini et al. 2012; Yang et al. 2014) and a uniform law.

1. Gravity law with an exponential distance decay function (`law = "GravExp"`). The arguments `mass_origin`, `mass_destination` (optional), `distance` and `param` will be used.
2. Normalized gravity law with an exponential distance decay function (`law = "NGravExp"`). The arguments `mass_origin`, `mass_destination` (optional), `distance` and `param` will be used.
3. Gravity law with a power distance decay function (`law = "GravPow"`). The arguments `mass_origin`, `mass_destination` (optional), `distance` and `param` will be used.
4. Normalized gravity law with a power distance decay function (`law = "NGravPow"`). The arguments `mass_origin`, `mass_destination` (optional), `distance` and `param` will be used.
5. Schneider's intervening opportunities law (`law = "Schneider"`). The arguments `mass_origin`, `mass_destination` (optional), `opportunity` and `param` will be used.
6. Radiation law (`law = "Rad"`). The arguments `mass_origin`, `mass_destination` (optional) and `opportunity` will be used.
7. Extended radiation law (`law = "RadExt"`). The arguments `mass_origin`, `mass_destination` (optional), `opportunity` and `param` will be used.
8. Uniform law (`law = "Unif"`). The argument `mass_origin` will be used to extract the number of locations.

Value

An object of class `TDLM`. A list of list of matrices containing for each parameter value the matrix of probabilities (called `proba`). If `length(param) = 1` or `law = "Rad"` or `law = "Unif` only a list of matrices will be returned.

Note

All the inputs should be based on the same number of locations sorted in the same order. It is recommended to use the location ID as vector names, matrix rownames and matrix colnames and to set `check_names = TRUE` to verify that everything is in order before running this function (`check_names = FALSE` by default). Note that the function `check_format_names()` can be used to control the validity of all the inputs before running the main package's functions.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

References

Lenormand M, Bassolas A, Ramasco JJ (2016). "Systematic comparison of trip distribution laws and models." *Journal of Transport Geography*, **51**, 158-169.

Carey HC (1858). *Principles of Social Science*. Lippincott.

Zipf GK (1946). "The P1 P2/D Hypothesis: On the Intercity Movement of Persons." *American Sociological Review*, **11**(6), 677–686.

Barthelemy M (2011). "Spatial Networks." *Physics Reports*, **499**, 1-101.

Schneider M (1959). "Gravity models and trip distribution theory." *Papers of the regional science association*, **5**, 51-58.

Simini F, González MC, Maritan A, Barabasi A (2012). "A universal model for mobility and migration patterns." *Nature*, **484**, 96-100.

Yang Y, Herrera C, Eagle N, González MC (2014). "Limits of Predictability in Commuting Flows in the Absence of Data for Calibration." *Scientific Reports*, **4**(5662), 5662.

See Also

`gof()` `run_law_model()` `run_model()` `extract_opportunities()` `check_format_names()`

Examples

```
data(mass)
data(distance)

mi <- as.numeric(mass[, 1])
mj <- mi

res <- run_law(
  law = "GravExp", mass_origin = mi, mass_destination = mj,
  distance = distance, opportunity = NULL, param = 0.01,
  check_names = FALSE
)

# print(res)
```

`run_law_model` *Estimate mobility flows based on different trip distribution laws and models*

Description

This function estimates mobility flows using different distribution laws and models. As described in Lenormand et al. (2016), the function uses a two-step approach to generate mobility flows by separating the trip distribution law, gravity or intervening opportunities, from the modeling approach used to generate the flows from this law.

Usage

```
run_law_model(
  law = "Unif",
  mass_origin,
  mass_destination = mass_origin,
  distance = NULL,
  opportunity = NULL,
  param = NULL,
  model = "UM",
  nb_trips = 1000,
  out_trips = NULL,
  in_trips = out_trips,
  average = FALSE,
  nbrep = 3,
  maxiter = 50,
  mindiff = 0.01,
  write_proba = FALSE,
  check_names = FALSE
)
```

Arguments

law	a character indicating which law to use (see Details).
mass_origin	a numeric vector representing the mass at origin (i.e. demand).
mass_destination	a numeric vector representing the mass at destination (i.e. attractiveness).
distance	a squared matrix representing the distance between locations (see Details).
opportunity	a squared matrix representing the number of opportunities between locations (see Details). Can be easily computed with extract_opportunities().
param	a vector of numeric value(s) used to adjust the importance of distance or opportunity associated with the chosen law. A single value or a vector of several parameter values can be used (see Return). Not necessary for the original radiation law or the uniform law (see Details).
model	a character indicating which model to use.
nb_trips	a numeric value indicating the total number of trips. Must be an integer if average = FALSE (see Details).
out_trips	a numeric vector representing the number of outgoing trips per location. Must be a vector of integers if average = FALSE (see Details).
in_trips	a numeric vector representing the number of incoming trips per location. Must be a vector of integers if average = FALSE (see Details).
average	a boolean indicating if the average mobility flow matrix should be generated instead of the nbrep matrices based on random draws (see Details).
nbrep	an integer indicating the number of replications associated to the model run. Note that nbrep = 1 if average = TRUE (see Details).

maxiter	an integer indicating the maximal number of iterations for adjusting the Doubly Constrained Model (see Details).
mindiff	a numeric strictly positive value indicating the stopping criterion for adjusting the Doubly Constrained Model (see Details).
write_proba	a boolean indicating if the estimation of the probability to move from one location to another obtained with the distribution law should be returned along with the flows estimations.
check_names	a boolean indicating if the ID location are used as vector names, matrix rownames and colnames and if they should be checked (see Note).

Details

First, we compute the matrix proba estimating the probability p_{ij} to observe a trip from location i to another location j ($\sum_i \sum_j p_{ij} = 1$). This probability is based on the demand m_i (argument mass_origin) and the attractiveness m_j (argument mass_destination). Note that the population is typically used as a surrogate for both quantities (this is why mass_destination = mass_origin by default). It also depends on the distance d_{ij} between locations (argument distance) OR the number of opportunities s_{ij} between locations (argument opportunity) depending on the chosen law. Both the effect of the distance and the number of opportunities can be adjusted with a parameter (argument param) except for the original radiation law and the uniform law.

In this package we consider eight probabilistic laws described in details in Lenormand et al. (2016). Four gravity laws (Carey 1858; Zipf 1946; Barthelemy 2011; Lenormand et al. 2016), three intervening opportunity laws (Schneider 1959; Simini et al. 2012; Yang et al. 2014) and a uniform law.

1. Gravity law with an exponential distance decay function (law = "GravExp"). The arguments mass_origin, mass_destination (optional), distance and param will be used.
2. Normalized gravity law with an exponential distance decay function (law = "NGravExp"). The arguments mass_origin, mass_destination (optional), distance and param will be used.
3. Gravity law with a power distance decay function (law = "GravPow"). The arguments mass_origin, mass_destination (optional), distance and param will be used.
4. Normalized gravity law with a power distance decay function (law = "NGravPow"). The arguments mass_origin, mass_destination (optional), distance and param will be used.
5. Schneider's intervening opportunities law (law = "Schneider"). The arguments mass_origin, mass_destination (optional), opportunity and param will be used.
6. Radiation law (law = "Rad"). The arguments mass_origin, mass_destination (optional) and opportunity will be used.
7. Extended radiation law (law = "RadExt"). The arguments mass_origin, mass_destination (optional), opportunity and param will be used.
8. Uniform law (law = "Unif"). The argument mass_origin will be used to extract the number of locations.

Second, we propose four constrained models to generate the flows from these distribution of probability. These models respect different level of constraints. These constraints can preserve the total number of trips (argument nb_trips) OR the number of out-going trips O_i (argument out_trips) AND/OR the number of in-coming D_j (argument in_trips) according to the model. The sum of out-going trips $\sum_i O_i$ should be equal to the sum of in-coming trips $\sum_j D_j$.

1. Unconstrained model (`model = "UM"`). Only `nb_trips` will be preserved (arguments `out_trips` and `in_trips` will not be used).
2. Production constrained model (`model = "PCM"`). Only `out_trips` will be preserved (arguments `nb_trips` and `in_trips` will not be used).
3. Attraction constrained model (`model = "ACM"`). Only `in_trips` will be preserved (arguments `nb_trips` and `out_trips` will not be used).
4. Doubly constrained model (`model = "DCM"`). Both `out_trips` and `in_trips` will be preserved (arguments `nb_trips`will not be used). The doubly constrained model is based on an Iterative Proportional Fitting process (Deming and Stephan 1940). The arguments `maxiter` (50 by default) and `mindiff` (0.01 by default) can be used to tune the model. `mindiff` is the minimal tolerated relative error between the simulated and observed marginals. `maxiter` ensures that the algorithm stops even if it has not converged toward the `mindiff` wanted value.

By default, when `average = FALSE`, `nbrep` matrices are generated from `proba` with multinomial random draws that will take different forms according to the model used. In this case, the models will deal with positive integers as inputs and outputs. Nevertheless, it is also possible to generate an average matrix based on a multinomial distribution (based on an infinite number of drawings). In this case, the models' inputs can be either positive integer or real numbers and the output (`nbrep = 1` in this case) will be a matrix of positive real numbers.

Value

An object of class `TDLM`. A list of list of matrices containing for each parameter value the `nbrep` simulated matrices and the matrix of probabilities (called `proba`) if `write_proba = TRUE`. If `length(param) = 1` or `law = "Rad"` or `law = "Unif` only a list of matrices will be returned.

Note

All the inputs should be based on the same number of locations sorted in the same order. It is recommended to use the location ID as vector names, matrix rownames and matrix colnames and to set `check_names = TRUE` to verify that everything is in order before running this function (`check_names = FALSE` by default). Note that the function `check_format_names()` can be used to control the validity of all the inputs before running the main package's functions.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

References

Lenormand M, Bassolas A, Ramasco JJ (2016). "Systematic comparison of trip distribution laws and models." *Journal of Transport Geography*, **51**, 158-169.

Carey HC (1858). *Principles of Social Science*. Lippincott.

Zipf GK (1946). "The P1 P2/D Hypothesis: On the Intercity Movement of Persons." *American Sociological Review*, **11**(6), 677–686.

Barthelemy M (2011). "Spatial Networks." *Physics Reports*, **499**, 1-101.

Schneider M (1959). "Gravity models and trip distribution theory." *Papers of the regional science association*, **5**, 51-58.

Simini F, González MC, Maritan A, Barabasi A (2012). "A universal model for mobility and migration patterns." *Nature*, **484**, 96-100.

Yang Y, Herrera C, Eagle N, González MC (2014). "Limits of Predictability in Commuting Flows in the Absence of Data for Calibration." *Scientific Reports*, **4**(5662), 5662.

Deming WE, Stephan FF (1940). "On a Least Squares Adjustment of a Sample Frequency Table When the Expected Marginal Totals Are Known." *Annals of Mathematical Statistics*, **11**, 427-444.

See Also

`gof()` `run_law()` `run_model()` `extract_opportunities()` `check_format_names()`

Examples

```
data(mass)
data(distance)

mi <- as.numeric(mass[, 1])
mj <- mi
Oi <- as.numeric(mass[, 2])
Dj <- as.numeric(mass[, 3])

res <- run_law_model(
  law = "GravExp", mass_origin = mi, mass_destination = mj,
  distance = distance, opportunity = NULL, param = 0.01,
  model = "DCM", nb_trips = NULL, out_trips = Oi, in_trips = Dj,
  average = FALSE, nbrep = 3, maxiter = 50, mindiff = 0.01,
  write_proba = FALSE,
  check_names = FALSE
)

print(res)
```

`run_model` *Estimate mobility flows based on different trip distribution models*

Description

This function estimates mobility flows using different distribution models. As described in Lenormand et al. (2016), we propose a two-step approach to generate mobility flows by separating the trip distribution law, gravity or intervening opportunities, from the modeling approach used to generate the flows from this law. This function only uses the second step to generate mobility flow based on a matrix of probabilities using different models.

Usage

```
run_model(
  proba,
  model = "UM",
  nb_trips = 1000,
  out_trips = NULL,
  in_trips = out_trips,
  average = FALSE,
  nbrep = 3,
  maxiter = 50,
  mindiff = 0.01,
  check_names = FALSE
)
```

Arguments

proba	a squared matrix of probability. The sum of the matrix element must be equal to 1. It will be normalized automatically if it is not the case.
model	a character indicating which model to use.
nb_trips	a numeric value indicating the total number of trips. Must be an integer if average = FALSE (see Details).
out_trips	a numeric vector representing the number of outgoing trips per location. Must be a vector of integers if average = FALSE (see Details).
in_trips	a numeric vector representing the number of incoming trips per location. Must be a vector of integers if average = FALSE (see Details).
average	a boolean indicating if the average mobility flow matrix should be generated instead of the nbrep matrices based on random draws (see Details).
nbrep	an integer indicating the number of replications associated to the model run. Note that nbrep = 1 if average = TRUE (see Details).
maxiter	an integer indicating the maximal number of iterations for adjusting the Doubly Constrained Model (see Details).
mindiff	a numeric strictly positive value indicating the stopping criterion for adjusting the Doubly Constrained Model (see Details).
check_names	a boolean indicating if the ID location are used as vector names, matrix row-names and colnames and if they should be checked (see Note).

Details

We propose four constrained models to generate the flow from the matrix of probabilities. These models respect different level of constraints. These constraints can preserve the total number of trips (argument nb_trips) OR the number of out-going trips O_i (argument out_trips) AND/OR the number of in-coming D_j (argument in_trips) according to the model. The sum of out-going trips $\sum_i O_i$ should be equal to the sum of in-coming trips $\sum_j D_j$.

1. Unconstrained model (model = "UM"). Only nb_trips will be preserved (arguments out_trips and in_trips will not be used).

2. Production constrained model (`model = "PCM"`). Only `out_trips` will be preserved (arguments `nb_trips` and `in_trips` will not be used).
3. Attraction constrained model (`model = "ACM"`). Only `in_trips` will be preserved (arguments `nb_trips` and `out_trips` will not be used).
4. Doubly constrained model (`model = "DCM"`). Both `out_trips` and `in_trips` will be preserved (arguments `nb_trips`will not be used). The doubly constrained model is based on an Iterative Proportional Fitting process (Deming and Stephan 1940). The arguments `maxiter` (50 by default) and `mindiff` (0.01 by default) can be used to tune the model. `mindiff` is the minimal tolerated relative error between the simulated and observed marginals. `maxiter` ensures that the algorithm stops even if it has not converged toward the `mindiff` wanted value.

By default, when `average = FALSE`, `nbrep` matrices are generated from `proba` with multinomial random draws that will take different forms according to the model used. In this case, the models will deal with positive integers as inputs and outputs. Nevertheless, it is also possible to generate an average matrix based on a multinomial distribution (based on an infinite number of drawings). In this case, the models' inputs can be either positive integer or real numbers and the output (`nbrep = 1` in this case) will be a matrix of positive real numbers.

Value

An object of class `TDLM`. A list of matrices containing the `nbrep` simulated matrices.

Note

All the inputs should be based on the same number of locations sorted in the same order. It is recommended to use the location ID as vector names, matrix rownames and matrix colnames and to set `check_names = TRUE` to verify that everything is in order before running this function (`check_names = FALSE` by default). Note that the function `check_format_names()` can be used to control the validity of all the inputs before running the main package's functions.

Author(s)

Maxime Lenormand (<maxime.lenormand@inrae.fr>)

References

Lenormand M, Bassolas A, Ramasco JJ (2016). "Systematic comparison of trip distribution laws and models." *Journal of Transport Geography*, **51**, 158-169.

Deming WE, Stephan FF (1940). "On a Least Squares Adjustment of a Sample Frequency Table When the Expected Marginal Totals Are Known." *Annals of Mathematical Statistics*, **11**, 427-444.

See Also

`gof()` `run_law_model()` `run_law()` `check_format_names()`

Examples

```
data(mass)
data(od)

proba <- od / sum(od)

Oi <- as.numeric(mass[, 2])
Dj <- as.numeric(mass[, 3])

res <- run_model(
  proba = proba,
  model = "DCM", nb_trips = NULL, out_trips = Oi, in_trips = Dj,
  average = FALSE, nbrep = 3, maxiter = 50, mindiff = 0.01,
  check_names = FALSE
)

# print(res)
```

Index