Package ‘intRinsic’

January 13, 2022

Title Likelihood-Based Intrinsic Dimension Estimators

Version 0.2.0

Maintainer Francesco Denti <francescodenti.personal@gmail.com>

Description Provides functions to estimate the intrinsic dimension of a
dataset via likelihood-based approaches. Specifically, the package
implements the 'TWO-NN' and 'Gride' estimators and the 'Hidalgo'
Bayesian mixture model. References:
Allegra et al. (2020, <doi:10.1038/s41598-020-72222-0>);
Denti et al. (2022+, <arXiv:2104.13832>);
Facco et al. (2017, <doi:10.1038/s41598-017-11873-y>);

License MIT + file LICENSE

URL https://github.com/Fradenti/intRinsic

Depends R (>= 3.6.0)
Imports dplyr, FNN, ggplot2, knitr, MCMCpack, pheatmap, Rcpp,
        reshape2, rlang, salso, stats, utils
LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes
ByteCompile true
Encoding UTF-8

RoxygenNote 7.1.2

Author Francesco Denti [aut, cre, cph]
        (<https://orcid.org/0000-0003-2978-4702>),
        Andrea Gilardi [aut] (<https://orcid.org/0000-0002-9424-7439>)

Repository CRAN

Date/Publication 2022-01-13 18:52:42 UTC

R topics documented:

        autoplot.gride_bayes ............................................ 2
Plot the simulated MCMC chains

Description

Use this method without the .gride_bayes suffix and after loading the ggplot2 package. It displays the traceplots of the chains generated with Metropolis-Hasting updates to visually assess mixing and convergence. Alternatively, it is possible to plot the posterior density.

Usage

```r
autoplot.gride_bayes(
  object,
  traceplot = FALSE,
  title = "Bayesian Gride - Posterior distribution",
  ...
)
```
**Arguments**

- **object**: object of class `gride_bayes`. It is obtained using the output of the `gride` function when `method = "bayes"`.
- **traceplot**: logical. If FALSE, the function returns a plot of the posterior density. If TRUE, the function returns the traceplots of the MCMC used to simulate from the posterior distribution.
- **title**: optional string to display as title.
- **...**: other arguments passed to specific methods.

**Value**

object of class `ggplot`. It could represent the traceplot of the posterior simulations for the Bayesian Gride model (traceplot = TRUE) or a density plot of the simulated posterior distribution (traceplot = FALSE).

**See Also**

- `gride`

---

**Description**

Use this method without the `.gride_evolution` suffix and after loading the `ggplot2` package. It plots the evolution of the Gride maximum likelihood estimates as a function of the average distance from the furthest NN of each point.

**Usage**

`autoplot.gride_evolution(object, title = "Gride Evolution", ...)`

**Arguments**

- **object**: an object of class `gride_evolution`.
- **title**: an optional string to customize the title of the plot.
- **...**: other arguments passed to specific methods.

**Value**

object of class `ggplot`. It displays the evolution of the Gride maximum likelihood estimates as a function of the average distance from `n2`.
**Description**

Use this method without the `.gride_mle` suffix and after loading the ggplot2 package. It display the density plot of sample obtained via parametric bootstrap for the Gride model.

**Usage**

```r
autoplot.gride_mle(object, title = "MLE Gride - Bootstrap sample", ...)```

**Arguments**

- `object`: object of class `gride_mle`. It is obtained using the output of the `gride` function when `method = "mle"`.
- `title`: title for the plot.
- `...`: other arguments passed to specific methods.

**Value**

object of class `ggplot`. It displays the density plot of the sample generated via parametric bootstrap to help the visual assessment of the uncertainty of the id estimates.

**See Also**

- `gride`

---

**Description**

Use this method without the `.Hidalgo` suffix and after loading the ggplot2 package. It produces several plots to explore the output of the Hidalgo model.

**Usage**

```r
autoplot.Hidalgo(  
  object,  
  type = c("raw_chains", "point_estimates", "class_plot", "clustering"),  
  class_plot_type = c("histogram", "density", "boxplot", "violin"),  
  class = NULL,  
  psm = NULL,  
  clust = NULL,  
  title = NULL,  
  ...  
)```

autoplot.Hidalgo

Arguments

object object of class Hidalgo, the output of the Hidalgo() function.
type character that indicates the type of plot that is requested. It can be:
"raw_chains" plot the MCMC and the ergodic means NOT corrected for label
switching;
"point_estimates" plot the posterior mean and median of the id for each ob-
servation, after the chains are processed for label switching;
"class_plot" plot the estimated id distributions stratified by the groups spec-
ified in the class vector;
"clustering" plot the posterior coclustering matrix. Rows and columns can
be stratified by and external class and/or a clustering solution.
class_plot_type
if type is chosen to be "class_plot", one can plot the stratified id estimates
with a "density" plot or a "histogram", or using "boxplots" or "violin" plots.
class factor variable used to stratify observations according to their the id estimates.
psm posterior similarity matrix containing the posterior probability of coclustering.
clust vector containing the cluster membership labels.
title character string used as title of the plot.
... other arguments passed to specific methods.

Value

a ggplot2 object produced by the function according to the type chosen. More precisely, if
method = "raw_chains" the function produces the traceplots of the parameters d_k, for k=1..K.
The ergodic means for all the chains are superimposed. The K chains that are plotted are not
post-processed. Ergo, they are subjected to label switching;
method = "point_estimates" the function returns two scatterplots displaying the posterior mean
and median id for each observation, after that the MCMC has been postprocessed to handle
label switching;
method = "class_plot" the function returns a plot that can be used to visually assess the relation
between the posterior id estimates and an external, categorical variable. The type of plot varies
according to the specification of class_plot_type, and it can be either a set of boxplots or
violin plots, or a collection of overlapping densities or histograms;
method = "clustering" the function displays the posterior similarity matrix, to allow the study
of the clustering structure present in the data estimated via the mixture model. Rows and
columns can be stratified by and external class and/or a clustering structure.

See Also

Hidalgo
autoplot.twonn_bayes  

Plot the output of the TWO-NN model estimated via the Bayesian approach

Description

Use this method without the .twonn_bayes suffix and after loading the ggplot2 package. The function returns the density plot of the posterior distribution computed with the bayes method.

Usage

autoplot.twonn_bayes(
  object,
  plot_low = 0,
  plot_upp = NULL,
  by = 0.05,
  title = "Bayesian TWO-NN",
  ...
)

Arguments

object  
object of class twonn_bayes, the output of the twonn function when method = "bayes".

plot_low  
lower bound of the interval on which the posterior density is plotted.

plot_upp  
upper bound of the interval on which the posterior density is plotted.

by  
step-size at which the sequence spanning the interval is incremented.

title  
character string used as title of the plot.

...  
other arguments passed to specific methods.

Value

ggplot2 object displaying the posterior distribution of the intrinsic dimension parameter.

See Also

twonn
**autoplot.twonn_linfit**

*Plot the output of the TWO-NN model estimated via least squares*

**Description**

Use this method without the `.twonn_linfit` suffix and after loading the `ggplot2` package. The function returns the representation of the linear regression that is fitted with the `linfit` method.

**Usage**

```r
autoplot.twonn_linfit(object, title = "TWO-NN Linear Fit", ...)
```

**Arguments**

- `object`: object of class `twonn_linfit`, the output of the `twonn` function when `method = "linfit"`.
- `title`: string used as title of the plot.
- `...`: other arguments passed to specific methods.

**Value**

A `ggplot2` object displaying the goodness of the linear fit of the TWO-NN model.

**See Also**

- `twonn`  

**autoplot.twonn_mle**

*Plot the output of the TWO-NN model estimated via the Maximum Likelihood approach*

**Description**

Use this method without the `.twonn_mle` suffix and after loading the `ggplot2` package. The function returns the point estimate along with the confidence bands computed via the `mle` method.

**Usage**

```r
autoplot.twonn_mle(object, title = "MLE TWO-NN", ...)
```

**Arguments**

- `object`: object of class `twonn_mle`, the output of the `twonn` function when `method = "mle"`.
- `title`: character string used as title of the plot.
- `...`: other arguments passed to specific methods.
compute_mus

Value
ggplot2 object displaying the point estimate and confidence interval obtained via maximum likelihood approach of the id parameter.

See Also
twonn

---

compute_mus | *Compute the ratio statistics needed for the intrinsic dimension estimation*

---

Description

The function compute_mus computes the ratios of distances between nearest neighbors (NNs) of generic order, denoted as $\mu(n_1,n_2)$. This quantity is at the core of all the likelihood-based methods contained in the package.

Usage

```r
compute_mus(X = NULL, dist_mat = NULL, n1 = 1, n2 = 2, Nq = FALSE, q = 3)
```

Arguments

- `X`  
a dataset with $n$ observations and $D$ variables.
- `dist_mat`  
a distance matrix computed between $n$ observations.
- `n1`  
order of the first NN considered. Default is 1.
- `n2`  
order of the second NN considered. Default is 2.
- `Nq`  
logical indicator. If TRUE, it provides the $N^q$ matrix needed for fitting the Hidalgo model.
- `q`  
integer, number of NN considered to build $N^q$.

Value

a vector containing the ratio statistics, an object of class `mus`. The length of the vector is equal to the number of observations considered, unless ties are present in the dataset. In that case, the duplicates are removed. Optionally, if Nq is TRUE, the function returns a list containing both the ratio statistics and the adjacency matrix $N^q$.

References


generalized_ratios_distribution

Examples

```r
data <- replicate(2, rnorm(1000))
mu <- compute_mus(data, n1 = 1, n2 = 2)
mudots <- compute_mus(data, n1 = 4, n2 = 8)
pre_hidalgo <- compute_mus(data, n1 = 4, n2 = 8, Nq = TRUE, q = 3)
```

---

**The Generalized Ratio distribution**

**Description**

Density function and random number generator for the Generalized Ratio distribution with NN orders equal to $n_1$ and $n_2$. See Denti et al., 2021+ for more details.

**Usage**

```r
rgera(nsim, n1 = 1, n2 = 2, d)
dgera(x, n1 = 1, n2 = 2, d, log = FALSE)
```

**Arguments**

- `nsim`: integer, the number of observations to generate.
- `n1`: order of the first NN considered. Default is 1.
- `n2`: order of the second NN considered. Default is 2.
- `d`: value of the intrinsic dimension.
- `x`: vector of quantiles.
- `log`: logical, if TRUE, it returns the log-density

**Value**

dgera gives the density. rgera returns a vector of random observations sampled from the generalized ratio distribution.

**References**


**Examples**

```r
draws <- rgera(100, 3, 5, 2)
density <- dgera(3, 3, 5, 2)
```
Description

The function can fit the Generalized ratios ID estimator under both the frequentist and the Bayesian frameworks, depending on the specification of the argument method. The model is the direct extension of the TWO-NN method presented in Facco et al., 2017. See also Denti et al., 2021+ for more details.

Usage

gride(
  X = NULL,
  dist_mat = NULL,
  mus_n1_n2 = NULL,
  method = c("mle", "bayes"),
  n1 = 1,
  n2 = 2,
  alpha = 0.95,
  nsim = 5000,
  upper_D = 50,
  burn_in = 2000,
  sigma = 0.5,
  start_d = NULL,
  a_d = 1,
  b_d = 1,
  ...
)

Arguments

X data matrix with n observations and D variables.
dist_mat distance matrix computed between the n observations.
mus_n1_n2 vector of generalized order NN distance ratios.
method the chosen estimation method. It can be
  "mle" maximum likelihood estimation;
  "bayes" estimation with the Bayesian approach.
n1 order of the first NN considered. Default is 1.
n2 order of the second NN considered. Default is 2.
alpha confidence level (for mle) or posterior probability in the credible interval (bayes).
nsim number of bootstrap samples or posterior simulation to consider.
upper_D nominal dimension of the dataset (upper bound for the maximization routine).
The function allows the study of the evolution of the \textit{id} estimates as a function of the scale of a dataset. A scale-dependent analysis is essential to identify the correct number of relevant directions in noisy data. To increase the average distance from the second NN (and thus the average neighborhood size) involved in the estimation, the function computes a sequence of \textit{Gr}ide models with increasing NN orders, \textit{n1} and \textit{n2}. See also Denti et al., 2021+ for more details.
Usage

```r
gride_evolution(X, vec_n1, vec_n2, upp_bound = 50)
```

Arguments

- **X**: data matrix with \( n \) observations and \( D \) variables.
- **vec_n1**: vector of integers, containing the smaller NN orders considered in the evolution.
- **vec_n2**: vector of integers, containing the larger NN orders considered in the evolution.
- **upp_bound**: upper bound for the interval used in the numerical optimization (via `optimize`). Default set to 50.

Value

- list containing the Gride evolution, the corresponding NN distance ratios, the average \( n_2 \)-th NN order distances, and the NN orders considered.

References


Examples

```r
X <- replicate(5,rnorm(10000,0,.1))
gride_evolution(X = X, vec_n1 = 2^(0:5), vec_n2 = 2^(1:6))
```

---

**Gibbs sampler for the Hidalgo model**

Description

The function fits the Heterogeneous intrinsic dimension algorithm, developed in Allegra et al., 2020. The model is a Bayesian mixture of Pareto distribution with modified likelihood to induce homogeneity across neighboring observations. The model can segment the observations into multiple clusters characterized by different intrinsic dimensions, which allows to capture hidden patterns in the data. For more details on the algorithm, refer to Allegra et al., 2020. For an example of application to basketball data, see Santos-Fernandez et al., 2021.
Usage

Hidalgo(
  X = NULL,
  dist_mat = NULL,
  K = 10,
  nsim = 5000,
  burn_in = 5000,
  thinning = 1,
  verbose = TRUE,
  q = 3,
  xi = 0.75,
  alpha_Dirichlet = 0.05,
  a0_d = 1,
  b0_d = 1,
  prior_type = c("Conjugate", "Truncated", "Truncated_PointMass"),
  D = NULL,
  pi_mass = 0.5
)

Arguments

X    data matrix with \( n \) observations and \( D \) variables.
dist_mat distance matrix computed between the \( n \) observations.
K    integer, number of mixture components.
nsim  number of MCMC iterations to run.
burn_in number of MCMC iterations to discard as burn-in period.
thinning integer indicating the thinning interval.
verbose logical, should the progress of the sampler be printed?
q    integer, first local homogeneity parameter. Default is 3.
xi    real between 0 and 1, second local homogeneity parameter. Default is 0.75.
alpha_Dirichlet parameter of the Dirichlet prior on the mixture weights. Default is 0.05, inducing a sparse mixture.
a0_d  shape parameter of the Gamma prior on \( d \).
b0_d  rate parameter of the Gamma prior on \( d \).
prior_type character, type of Gamma prior on \( d \), can be
  "Conjugate" a conjugate Gamma distribution is elicited;
  "Truncated" the conjugate Gamma prior is truncated over the interval \((0,D)\);
  "Truncated_PointMass" same as "Truncated", but a point mass is placed on \( D \) to allow the id to be identically equal to the nominal dimension.
D    integer, the maximal dimension of the dataset.
pi_mass probability placed a priori on \( D \) when Truncated_PointMass is chosen.
Value

object of class Hidalgo, which is a list containing

cluster_prob chains of the posterior mixture weights;
member_labels chains of the membership labels for all the observations;
id_raw chains of the K intrinsic dimensions parameters, one per mixture component;
id_postpr a chain for each observation, corrected for label switching;
id_summary a matrix containing, for each observation, the value of posterior mean and the 5%, 25%, 50%, 75%, 95% quantiles;
recap a list with the objects and specifications passed to the function used in the estimation.

References


See Also

id_by_class and psm_and_cluster to understand how to further postprocess the results.

Examples

```r
X <- replicate(5,rnorm(500))
X[1:250,1:2] <- 0
X[1:250,] <- X[1:250,] + 4
oracle <- rep(1:2,rep(250,2))
# this is just a short example
# increase the number of iterations to improve mixing and convergence
h_out <- Hidalgo(X, nsim = 500, burn_in = 500)
id_by_class(h_out, oracle)
```

id_by_class

Stratification of the id by an external categorical variable

Description

The function computes summary statistics (mean, median, and standard deviation) of the postprocessed chains of the intrinsic dimension stratified by an external categorical variable.
Usage

id_by_class(object, class)

Arguments

object object of class Hidalgo, the output of the Hidalgo() function.
class factor according to the observations should be stratified by.

Value

a data.frame containing the posterior id means, medians, and standard deviations stratified by the levels of the variable class.

See Also

Hidalgo

Examples

X <- replicate(5, rnorm(500))
X[1:250, 1:2] <- 0
oracle <- rep(1:2, rep(250, 2))
h_out <- Hidalgo(X)
id_by_class(h_out, oracle)

Description

Print Gride Bayes object

Usage

## S3 method for class 'gride_bayes'
print(x, ...)

Arguments

x object of class gride_bayes(), obtained from the function gride_bayes().
...

Value

the function prints a summary of the Bayesian Grid to console.
print.gride_evolution  

Print Gride evolution object

Description
Print Gride evolution object

Usage
```r
## S3 method for class 'gride_evolution'
print(x, ...)
```

Arguments
- `x` object of class `gride_evolution`, obtained from the function `gride_evolution()`.
- `...` ignored.

Value
the function prints a summary of the Gride evolution to console.

print.gride_mle  

Print Gride MLE object

Description
Print Gride MLE object

Usage
```r
## S3 method for class 'gride_mle'
print(x, ...)
```

Arguments
- `x` object of class `gride_mle`, obtained from the function `gride_mle()`.
- `...` ignored.

Value
the function prints a summary of the Gride estimated via maximum likelihood to console.
## print.Hidalgo

**Print the Hidalgo object**

**Description**

Print the Hidalgo object

**Usage**

```r
## S3 method for class 'Hidalgo'
print(x, ...)
```

**Arguments**

- `x`: an object of class Hidalgo, obtained from the function Hidalgo().
- `...`: other arguments passed to specific methods.

**Value**

the function prints a summary of the Hidalgo MCMC run to console.

---

## print.hidalgo_psm

**Print the summary of the clustering solution**

**Description**

Print the summary of the clustering solution

**Usage**

```r
## S3 method for class 'hidalgo_psm'
print(x, ...)
```

**Arguments**

- `x`: object of class hidalgo_psm, obtained from the function psm_and_cluster().
- `...`: ignored.

**Value**

the function prints a summary of the clustering solution to console.
print.mus  

Print the ratio statistics output

Description
Print the ratio statistics output

Usage
```r
## S3 method for class 'mus'
print(x, ...)
```

Arguments
- `x` object of class `mus`, obtained from the function `compute_mus()`.  
- `...` ignored.

Value
the function prints a summary of the computed ratio statistics to console.

print.twonn_bayes  

Print TWO-NN Bayes object

Description
Print TWO-NN Bayes object

Usage
```r
## S3 method for class 'twonn_bayes'
print(x, ...)
```

Arguments
- `x` object of class `twonn_bayes`, obtained from the function `twonn_bayes()`.  
- `...` ignored.

Value
the function prints a summary of the Bayesian TWO-NN to console.
print.twonn_dec_by

---

**print.twonn_dec_by**  
*Print TWO-NN evolution object decimated via halving steps*

---

**Description**

Print TWO-NN evolution object decimated via halving steps

**Usage**

```r
## S3 method for class 'twonn_dec_by'
print(x, ...)
```

**Arguments**

- `x`  
  object of class `twonn_dec_prop`, obtained from the function `twonn_dec_prop()`.  
- `...`  
  ignored.

**Value**

the function prints a summary of the decimated TWO-NN to console.

---

print.twonn_dec_prop

---

**print.twonn_dec_prop**  
*Print TWO-NN evolution object decimated via vector of proportions*

---

**Description**

Print TWO-NN evolution object decimated via vector of proportions

**Usage**

```r
## S3 method for class 'twonn_dec_prop'
print(x, ...)
```

**Arguments**

- `x`  
  object of class `twonn_dec_prop`, obtained from the function `twonn_dec_prop()`.  
- `...`  
  ignored.

**Value**

the function prints a summary of the decimated TWO-NN to console.
### Description

Print TWO-NN Least Squares output

### Usage

```r
## S3 method for class 'twonn_linfit'
print(x, ...)
```

### Arguments

- `x`: object of class `twonn_linfit`, obtained from the function `twonn_linfit()`.
- `...`: ignored.

### Value

the function prints a summary of the TWO-NN estimated via least squares to console.

---

### Description

Print TWO-NN MLE output

### Usage

```r
## S3 method for class 'twonn_mle'
print(x, ...)
```

### Arguments

- `x`: object of class `twonn_mle`, obtained from the function `twonn_mle()`.
- `...`: ignored.

### Value

the function prints a summary of the TWO-NN estimated via maximum likelihood to console.
Description

The function computes the posterior similarity (coclustering) matrix (psm) and estimates a representative partition of the observations from the MCMC output. The user can provide the desired number of clusters, or estimate a partition minimizing a loss function on the space of the partitions. In the latter case, function uses the package salso (Dahl et al., 2021+), that the user needs to load.

Usage

```r
psm_and_cluster(
  object,
  clustering_method = c("dendrogram", "salso"),
  K = 2,
  nCores = 1,
  ...
)
```

Arguments

- `object` object of class Hidalgo, the output of the Hidalgo function.
- `clustering_method` character indicating the method to use to perform clustering. It can be
  - "dendrogram" thresholding the adjacency dendrogram with a given number (K);
  - "salso" estimation via minimization of several partition estimation criteria. The default loss function is the variation of information.
- `K` number of clusters to recover by thresholding the dendrogram obtained from the psm.
- `nCores` parameter for the salso function: the number of CPU cores to use. A value of zero indicates to use all cores on the system.
- `...` optional additional parameter to pass to salso().

Value

list containing the posterior similarity matrix (psm) and the estimated partition clust.

References

### Swissroll

**Generates a noise-free Swiss roll dataset**

### Description

The function creates a three-dimensional dataset with coordinates following the Swiss roll mapping, transforming random uniform data points sampled on the interval $(0, 10)$.

### Usage

```r
Swissroll(n)
```

### Arguments

- `n` number of observations contained in the output dataset.

### Value

A three-dimensional data frame containing the coordinates of the points generated via the Swiss roll mapping.

### Examples

```r
Data <- Swissroll(1000)
```
twonn

**TWO-NN estimator**

**Description**

The function can fit the two-nearest neighbor estimator within the maximum likelihood and the Bayesian frameworks. Also, one can obtain the estimates using least squares estimation, depending on the specification of the argument `method`. This model has been originally presented in Facco et al., 2017. See also Denti et al., 2021+ for more details.

**Usage**

```r
twonn(
    X = NULL,
    dist_mat = NULL,
    mus = NULL,
    method = c("mle", "linfit", "bayes"),
    alpha = 0.95,
    c_trimmed = 0.01,
    unbiased = TRUE,
    a_d = 0.001,
    b_d = 0.001,
    ...
)
```

**Arguments**

- **X** data matrix with \( n \) observations and \( D \) variables.
- **dist_mat** distance matrix computed between the \( n \) observations.
- **mus** vector of second to first NN distance ratios.
- **method** chosen estimation method. It can be "mle" for maximum likelihood estimator; "linfit" for estimation via the least squares approach; "bayes" for estimation with the Bayesian approach.
- **alpha** the confidence level (for `mle` and least squares fit) or posterior probability in the credible interval (bayes).
- **c_trimmed** the proportion of trimmed observations.
- **unbiased** logical, applicable when `method = "mle"`. If `TRUE`, the MLE is corrected to ensure unbiasedness.
- **a_d** shape parameter of the Gamma prior on the parameter \( d \), applicable when `method = "bayes"`.
- **b_d** rate parameter of the Gamma prior on the parameter \( d \), applicable when `method = "bayes"`.
- **...** additional arguments for the different methods.
twonn_decimated

Value

list characterized by a class type that depends on the method chosen. Regardless the method, the output list always contains the object est, which provides the estimated intrinsic dimension along with uncertainty quantification. The remaining objects varies with the estimation method. In particular, if

- method = "mle" the output reports the MLE and the relative confidence interval;
- method = "linfit" the output includes the lm() object used for the computation;
- method = "bayes" the output contains the \((1 + \alpha) / 2\) and \((1 - \alpha) / 2\) quantiles, mean, mode and median of the posterior distribution of \(d\).

References


Examples

```r
# dataset with 1000 observations and id = 2
X <- replicate(2,rnorm(1000))
twonn(X)

# dataset with 1000 observations and id = 3
Y <- replicate(3,runif(1000))
twonn(Y)

# Bayesian and least squares estimate from distance matrix
dm <- as.matrix(dist(Y,method = "manhattan"))
twonn(dist_mat = dm,method = "bayes")
twonn(dist_mat = dm,method = "linfit")
```

twonn_decimated

Estimate the decimated TWO-NN evolution with halving steps or vector of proportions

Description

The estimation of the id is related to the scale of the dataset. To escape the local reach of the TWO-NN estimator, Facco et al. (2017) proposed to subsample the original dataset in order to induce greater distances between the data points. By investigating the estimates’ evolution as a function of the size of the neighborhood, it is possible to obtain information about the validity of the modeling assumptions and the robustness of the model in the presence of noise.
**twonn_decimated**

**Usage**

```r
twonn_decimated(
  X,
  method = c("steps", "proportions"),
  steps = 0,
  proportions = 1,
  seed = NULL
)
```

**Arguments**

- **X**
  - data matrix with n observations and D variables.
- **method**
  - method to use for decimation:
  - "steps" the number of times the dataset is halved;
  - "proportion" the dataset is subsampled according to a vector of proportions.
- **steps**
  - number of times the dataset is halved.
- **proportions**
  - vector containing the fractions of the dataset to be considered.
- **seed**
  - random seed controlling the sequence of sub-sampled observations.

**Value**

list containing the two-NN evolution (maximum likelihood estimation and confidence intervals), the average distance from the second NN, and the vector of proportions that were considered. According to the chosen estimation method, it is accompanied with the vector of proportions or halving steps considered.

**References**


**See Also**

- `twonn`

**Examples**

```r
X <- replicate(4, rnorm(1000))
twonn_decimated(X, method = "proportions",
                proportions = c(1, .5, .2, .1, .01))
```
Index

autoplot.gride_bayes, 2
autoplot.gride_evolution, 3
autoplot.gride_mle, 4
autoplot.Hidalgo, 4
autoplot.twonn_bayes, 6
autoplot.twonn_linfit, 7
autoplot.twonn_mle, 7

compute_mus, 8
dgera
   (generalized_ratios_distribution), 9
generalized_ratios_distribution, 9
ggplot, 3, 4
ggplot2, 5–8
gride, 3, 4, 10
gride_evolution, 11
Hidalgo, 5, 12, 15, 22
id_by_class, 14, 14

print.gride_bayes, 15
print.gride_evolution, 16
print.gride_mle, 16
print.Hidalgo, 17
print.hidalgo_psm, 17
print.mus, 18
print.twonn_bayes, 18
print.twonn_dec_by, 19
print.twonn_dec_prop, 19
print.twonn_linfit, 20
print.twonn_mle, 20
psm_and_cluster, 14, 21

ergera
   (generalized_ratios_distribution), 9

salos, 22
Swissroll, 22
twonn, 6–8, 23, 25
twonn_decimated, 24