Package: intRinsic (via r-universe)

October 26, 2026

Title Likelihood-Based Intrinsic Dimension Estimators

Version 1.1.2

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. In addition, the first reference contains an extended vignette on the usage of the 'TWO-NN' and 'Hidalgo' models. References: Denti (2023, <doi:10.18637/jss.v106.i09>); Allegra et al. (2020, <doi:10.1038/s41598-020-72222-0>); Denti et al. (2022, <doi:10.1038/s41598-022-20991-1>); Facco et al. (2017, <doi:10.1038/s41598-017-11873-y>); Santos-Fernandez et al. (2021, <doi:10.1038/s41598-022-20991-1>).

License MIT + file LICENSE

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

BugReports https://github.com/fradenti/intRinsic/issues

Depends R (>= 4.2.0)

Imports dplyr, FNN, ggplot2, knitr, Rcpp, reshape2, rlang, stats, utils, salso

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

ByteCompile true

Encoding UTF-8

RoxygenNote 7.3.3

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

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

Date/Publication 2025-12-17 14:40:08 UTC

RemoteUrl https://github.com/cran/intRinsic

RemoteRef HEAD

RemoteSha ecf4aa20b9030666171940917e4186bd6be34f18

Contents

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 . . . 8
auxHidalgo . . . 8
clustering . . . 9
compute_mus . . . 11
generalized_ratios_distribution . . . 12
gride . . . 13
gride_evolution . . . 15
Hidalgo . . . 16
id_by_class . . . 19
Swissroll . . . 20
twonn . . . 20
twonn_decimated . . . 23
twonn_decimation . . . 24

Index **26**

`autoplot.gride_bayes` *Plot the simulated MCMC chains for the Bayesian* `Gride`

Description

Use this method without the `.gride_bayes` suffix. It displays the traceplot of the chain generated with Metropolis-Hasting updates to visually assess mixing and convergence. Alternatively, it is possible to plot the posterior density.

Usage

```
## S3 method for class 'gride_bayes'
autoplot(
  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`

Other autoplot methods: `autoplot.Hidalgo()`, `autoplot.twonn_bayes()`, `autoplot.twonn_linfit()`, `autoplot.twonn_mle()`

`autoplot.gride_evolution` *Plot the evolution of* `Gride` *estimates*

Description

Use this method without the `.gride_evolution` suffix. It plots the evolution of the `id` estimates as a function of the average distance from the furthest NN of each point.

Usage

```
## S3 method for class 'gride_evolution'
autoplot(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 the evolution of the Gride maximum likelihood estimates as a function of the average distance from `n2`.

autoplot.gride_mle *Plot the simulated bootstrap sample for the MLE* Gride

Description

Use this method without the .gride_mle suffix. It displays the density plot of sample obtained via parametric bootstrap for the Gride model.

Usage

```
## S3 method for class 'gride_mle'
autoplot(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

autoplot.Hidalgo *Plot the output of the* Hidalgo *function*

Description

Use this method without the .Hidalgo suffix. It produces several plots to explore the output of the Hidalgo model.

Usage

```
## S3 method for class 'Hidalgo'
autoplot(
  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,
  ...
)
```

Arguments

`object`	object of class `Hidalgo`, the output of the `Hidalgo()` function.
`type`	character that indicates the requested type of plot. 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 observation, after the chains are processed for label switching; `"class_plot"` plot the estimated id distributions stratified by the groups specified in the class vector; `"clustering"` plot the posterior coclustering matrix. Rows and columns can be stratified by an exogenous 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 functions 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 relationship 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 an exogenous class and/or a clustering structure.

See Also

`Hidalgo`

Other autoplot methods: `autoplot.gride_bayes()`, `autoplot.twonn_bayes()`, `autoplot.twonn_linfit()`, `autoplot.twonn_mle()`

`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. The function returns the density plot of the posterior distribution computed with the `bayes` method.

Usage

```
## S3 method for class 'twonn_bayes'
autoplot(
  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

Other autoplot methods: autoplot.Hidalgo(), autoplot.gride_bayes(), autoplot.twonn_linfit(), autoplot.twonn_mle()

---

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

Description

Use this method without the .twonn_linfit suffix. The function returns the representation of the linear regression that is fitted with the linfit method.

Usage

```
## S3 method for class 'twonn_linfit'
autoplot(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

Other autoplot methods: autoplot.Hidalgo(), autoplot.gride_bayes(), autoplot.twonn_bayes(), autoplot.twonn_mle()

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. The function returns the point estimate along with the confidence bands computed via the mle method.

Usage

```
## S3 method for class 'twonn_mle'
autoplot(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.

Value

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

See Also

twonn

Other autoplot methods: autoplot.Hidalgo(), autoplot.gride_bayes(), autoplot.twonn_bayes(), autoplot.twonn_linfit()

auxHidalgo *Auxiliary functions for the* Hidalgo *model*

Description

Collection of functions used to extract meaningful information from the object returned by the function Hidalgo

Usage

```
posterior_means(x)

initial_values(x)

posterior_medians(x)

credible_intervals(x, alpha = 0.95)
```

Arguments

x	object of class Hidalgo, the output of the Hidalgo() function.
alpha	posterior probability contained in the computed credible interval.

Value

posterior_mean returns the observation-specific id posterior means estimated with Hidalgo.

initial_values returns a list with the parameter specification passed to the model.

posterior_median returns the observation-specific id posterior medians estimated with Hidalgo.

credible_interval returns the observation-specific credible intervals for a specific probability alpha.

---

clustering *Posterior similarity matrix and partition estimation*

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 optimal clustering solution by minimizing a loss function on the space of the partitions. In the latter case, the function uses the package salso (Dahl et al., 2021), that the user needs to load.

Usage

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

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

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

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.
...	ignored.
x	object of class hidalgo_psm, obtained from the function clustering().

Value

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

References

D. B. Dahl, D. J. Johnson, and P. Müller (2022), "Search Algorithms and Loss Functions for Bayesian Clustering", Journal of Computational and Graphical Statistics, doi:10.1080/10618600.2022.2069779.

David B. Dahl, Devin J. Johnson and Peter Müller (2022). "salso: Search Algorithms and Loss Functions for Bayesian Clustering". R package version 0.3.0. https://CRAN.R-project.org/package=salso

See Also

Hidalgo, salso

Examples

```
library(salso)
X            <- replicate(5,rnorm(500))
X[1:250,1:2] <- 0
h_out        <- Hidalgo(X)
clustering(h_out)
```

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

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

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

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

## S3 method for class 'mus'
plot(x, range_d = NULL, ...)
```

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.
x	object of class mus, obtained from the function compute_mus().
...	ignored.
range_d	a sequence of values for which the generalized ratios density is superimposed to the histogram of mus.

Value

the principal output of this function is 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 an object of class mus_Nq, a list containing both the ratio statistics mus and the adjacency matrix NQ.

References

Facco E, D'Errico M, Rodriguez A, Laio A (2017). "Estimating the intrinsic dimension of datasets by a minimal neighborhood information." Scientific Reports, 7(1). ISSN 20452322, doi:10.1038/s4159801711873y.

Denti F, Doimo D, Laio A, Mira A (2022). "The generalized ratios intrinsic dimension estimator." Scientific Reports, 12(20005). ISSN 20452322, doi:10.1038/s41598022209911.

Examples

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

`generalized_ratios_distribution`

The Generalized Ratio distribution

Description

Density function and random number generator for the Generalized Ratio distribution with NN orders equal to `n1` and `n2`. See Denti et al., 2022 for more details.

Usage

```
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

Denti F, Doimo D, Laio A, Mira A (2022). "The generalized ratios intrinsic dimension estimator." Scientific Reports, 12(20005). ISSN 20452322, doi:10.1038/s41598022209911.

Examples

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

---

gride Gride*: the Generalized Ratios ID Estimator*

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., 2022 \ 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,
  ...
)

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

## S3 method for class 'gride_bayes'
summary(object, ...)

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

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

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

## S3 method for class 'gride_mle'
summary(object, ...)

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

## S3 method for class 'gride_mle'
plot(x, ...)
```

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).
burn_in	number of iterations to discard from the MCMC sample. Applicable if method = "bayes".
sigma	standard deviation of the Gaussian proposal used in the MH step. Applicable if method = "bayes".
start_d	initial value for the MCMC chain. If NULL, the MLE is used. Applicable if method = "bayes".
a_d	shape parameter of the Gamma prior distribution for d. Applicable if method = "bayes".
b_d	rate parameter of the Gamma prior distribution for d. Applicable if method = "bayes".
...	other arguments passed to specific methods.
x	object of class gride_mle. It is obtained using the output of the gride function when method = "mle".
object	object of class gride_mle, obtained from the function gride_mle().

Value

a list containing the id estimate obtained with the Gride method, along with the relative confidence or credible interval (object est). The class of the output object changes according to the chosen method. Similarly, the remaining elements stored in the list reports a summary of the key quantities involved in the estimation process, e.g., the NN orders n1 and n2.

References

Facco E, D'Errico M, Rodriguez A, Laio A (2017). "Estimating the intrinsic dimension of datasets by a minimal neighborhood information." Scientific Reports, 7(1). ISSN 20452322, doi:10.1038/s4159801711873y.

Denti F, Doimo D, Laio A, Mira A (2022). "The generalized ratios intrinsic dimension estimator." Scientific Reports, 12(20005). ISSN 20452322, doi:10.1038/s41598022209911.

Examples

```
X  <- replicate(2,rnorm(500))
dm <- as.matrix(dist(X,method = "manhattan"))
res <- gride(X, nsim = 500)
res
plot(res)
gride(dist_mat = dm, method = "bayes", upper_D =10,
nsim = 500, burn_in = 100)
```

`gride_evolution` Gride *evolution based on Maximum Likelihood Estimation*

Description

The function allows the study of the evolution of the `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 `Gride` models with increasing NN orders, `n1` and `n2`. See also Denti et al., 2022 for more details.

Usage

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

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

## S3 method for class 'gride_evolution'
plot(x, ...)
```

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 is set to 50.
`x`	an object of class `gride_evolution`.
`...`	other arguments passed to specific methods.

Value

list containing the Gride evolution, the corresponding NN distance ratios, the average n2-th NN order distances, and the NN orders considered.

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

References

Denti F, Doimo D, Laio A, Mira A (2022). "The generalized ratios intrinsic dimension estimator." Scientific Reports, 12(20005). ISSN 20452322, doi:10.1038/s41598022209911.

Examples

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

`Hidalgo` *Fit 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. This permits 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
```

```
)

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

## S3 method for class 'Hidalgo'
plot(x, type = c("A", "B", "C"), class = NULL, ...)

## S3 method for class 'Hidalgo'
summary(object, ...)

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

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 number between 0 and 1, second local homogeneity parameter. Default is 0.75.
alpha_Dirichlet	parameter of the symmetric Dirichlet prior on the mixture weights. Default is 0.05, inducing a sparse mixture. Values that are too small (i.e., lower than 0.005) may cause underflow.
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.
x	object of class Hidalgo, the output of the Hidalgo() function.
...	other arguments passed to specific methods.
type	character that indicates the type of plot that is requested. It can be: "A" plot the MCMC and the ergodic means NOT corrected for label switching;

"B" plot the posterior mean and median of the id for each observation, after the chains are processed for label switching;

"C" plot the estimated id distributions stratified by the groups specified in the class vector;

class factor variable used to stratify observations according to their the id estimates.

object object of class Hidalgo, the output of the Hidalgo() function.

Value

object of class Hidalgo, which is a list containing

cluster_prob chains of the posterior mixture weights;

membership_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

Allegra M, Facco E, Denti F, Laio A, Mira A (2020). "Data segmentation based on the local intrinsic dimension." Scientific Reports, 10(1), 1–27. ISSN 20452322, doi:10.1038/s41598020722220,

Santos-Fernandez E, Denti F, Mengersen K, Mira A (2021). "The role of intrinsic dimension in high-resolution player tracking data – Insights in basketball." Annals of Applied Statistics - Forthcoming, – ISSN 2331-8422, 2002.04148, doi:10.1038/s41598022209911

See Also

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

Examples

```
set.seed(1234)
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)
plot(h_out, type =  "B")
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 post-processed chains of the intrinsic dimension stratified by an external categorical variable.

Usage

```
id_by_class(object, class)

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

Arguments

`object`	object of class `Hidalgo`, the output of the `Hidalgo()` function.
`class`	factor according to the observations should be stratified by.
`x`	object of class `hidalgo_class`, the output of the `id_by_class()` function.
`...`	other arguments passed to specific methods.

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)
```

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

```
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

```
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., 2022 for more details.

Usage

```
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,
  ...
)

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

## S3 method for class 'twonn_bayes'
summary(object, ...)

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

## S3 method for class 'twonn_bayes'
plot(x, plot_low = 0.001, plot_upp = NULL, by = 0.05, ...)

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

## S3 method for class 'twonn_linfit'
summary(object, ...)

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

## S3 method for class 'twonn_linfit'
plot(x, ...)

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

## S3 method for class 'twonn_mle'
summary(object, ...)

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

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

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"`.
`...`	ignored.
`x`	object of class `twonn_mle`, the output of the `twonn` function when `method = "mle"`.
`object`	object of class `twonn_mle`, obtained from the function `twonn_mle()`.
`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.

Value

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

Facco E, D'Errico M, Rodriguez A, Laio A (2017). "Estimating the intrinsic dimension of datasets by a minimal neighborhood information." Scientific Reports, 7(1). ISSN 20452322, doi:10.1038/s4159801711873y.

Denti F, Doimo D, Laio A, Mira A (2022). "The generalized ratios intrinsic dimension estimator." Scientific Reports, 12(20005). ISSN 20452322, doi:10.1038/s41598022209911.

Examples

```
# 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))
#  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.

Usage

```
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

Facco E, D'Errico M, Rodriguez A, Laio A (2017). "Estimating the intrinsic dimension of datasets by a minimal neighborhood information." Scientific Reports, 7(1). ISSN 20452322, doi:10.1038/s4159801711873y.

Denti F, Doimo D, Laio A, Mira A (2022). "The generalized ratios intrinsic dimension estimator." Scientific Reports, 12(20005). ISSN 20452322, doi:10.1038/s41598022209911.

See Also

twonn

`twonn_decimation` *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.

Usage

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

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

## S3 method for class 'twonn_dec_prop'
plot(x, CI = FALSE, proportions = FALSE, ...)

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

## S3 method for class 'twonn_dec_by'
plot(x, CI = FALSE, steps = FALSE, ...)
```

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	logical, if TRUE, the x-axis reports the number of halving steps. If FALSE, the x-axis reports the log10 average distance.
proportions	logical, if TRUE, the x-axis reports the number of decimating proportions. If FALSE, the x-axis reports the log10 average distance.
seed	random seed controlling the sequence of sub-sampled observations.
x	object of class twonn_dec_prop, obtained from the function twonn_dec_prop().
...	ignored.
CI	logical, if TRUE, the confidence intervals are plotted

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

Facco E, D'Errico M, Rodriguez A, Laio A (2017). "Estimating the intrinsic dimension of datasets by a minimal neighborhood information." Scientific Reports, 7(1). ISSN 20452322, doi:10.1038/s4159801711873y.

Denti F, Doimo D, Laio A, Mira A (2022). "The generalized ratios intrinsic dimension estimator." Scientific Reports, 12(20005). ISSN 20452322, doi:10.1038/s41598022209911.

See Also

twonn

Examples

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

Index

∗ **autoplot methods**
  autoplot.gride_bayes, 2
  autoplot.Hidalgo, 4
  autoplot.twonn_bayes, 6
  autoplot.twonn_linfit, 7
  autoplot.twonn_mle, 8

autoplot.gride_bayes, 2, *6–8*
autoplot.gride_evolution, 3
autoplot.gride_mle, 4
autoplot.Hidalgo, *3*, 4, *7, 8*
autoplot.twonn_bayes, *3*, 6, *6–8*
autoplot.twonn_linfit, *3*, *6*, 7, *7, 8*
autoplot.twonn_mle, *3*, *6, 7*, 8
auxHidalgo, 8

clustering, 9, *18*
compute_mus, 11
credible_intervals *(*auxHidalgo*)*, 8

dgera
        *(*generalized_ratios_distribution*)*, 12

generalized_ratios_distribution, 12
ggplot, *3, 4*
ggplot2, *5*, *7, 8*
gride, *3, 4*, 13
gride_evolution, 15

Hidalgo, *6*, *10*, 16, *19*

id_by_class, *18*, 19
initial_values *(*auxHidalgo*)*, 8

plot.gride_bayes *(*gride*)*, 13
plot.gride_evolution *(*gride_evolution*)*, 15
plot.gride_mle *(*gride*)*, 13
plot.Hidalgo *(*Hidalgo*)*, 16
plot.hidalgo_psm *(*clustering*)*, 9
plot.mus *(*compute_mus*)*, 11
plot.twonn_bayes *(*twonn*)*, 20
plot.twonn_dec_by *(*twonn_decimation*)*, 24
plot.twonn_dec_prop *(*twonn_decimation*)*, 24
plot.twonn_linfit *(*twonn*)*, 20
plot.twonn_mle *(*twonn*)*, 20
posterior_means *(*auxHidalgo*)*, 8
posterior_medians *(*auxHidalgo*)*, 8
print.gride_bayes *(*gride*)*, 13
print.gride_evolution *(*gride_evolution*)*, 15
print.gride_mle *(*gride*)*, 13
print.Hidalgo *(*Hidalgo*)*, 16
print.hidalgo_class *(*id_by_class*)*, 19
print.hidalgo_psm *(*clustering*)*, 9
print.mus *(*compute_mus*)*, 11
print.mus_Nq *(*compute_mus*)*, 11
print.summary.gride_bayes *(*gride*)*, 13
print.summary.gride_mle *(*gride*)*, 13
print.summary.Hidalgo *(*Hidalgo*)*, 16
print.summary.twonn_bayes *(*twonn*)*, 20
print.summary.twonn_linfit *(*twonn*)*, 20
print.summary.twonn_mle *(*twonn*)*, 20
print.twonn_bayes *(*twonn*)*, 20
print.twonn_dec_by *(*twonn_decimation*)*, 24
print.twonn_dec_prop *(*twonn_decimation*)*, 24
print.twonn_linfit *(*twonn*)*, 20
print.twonn_mle *(*twonn*)*, 20

rgera
        *(*generalized_ratios_distribution*)*, 12

salso, *10*
summary.gride_bayes *(*gride*)*, 13
summary.gride_mle *(*gride*)*, 13
summary.Hidalgo *(*Hidalgo*)*, 16

summary.twonn_bayes *(*twonn*)*, 20
summary.twonn_linfit *(*twonn*)*, 20
summary.twonn_mle *(*twonn*)*, 20
Swissroll, 20

twonn, *7*, *8*, 20, *24*, *25*
twonn_decimated, 23
twonn_decimation, 24