Package ‘kldest’

April 9, 2024

Type Package

Title Sample-Based Estimation of Kullback-Leibler Divergence

Version 1.0.0

Maintainer Niklas Hartung <niklas.hartung@gmail.com>

Description Estimation algorithms for Kullback-Leibler divergence between two probability distributions, based on one or two samples, and including uncertainty quantification. Distributions can be uni- or multivariate and continuous, discrete or mixed.

Encoding UTF-8

RoxygenNote 7.2.3

Imports stats, RANN

Suggests knitr, rmarkdown, KernSmooth, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/Needs/website ggplot2, reshape2, MASS

URL https://niklhart.github.io/kldest/

BugReports https://github.com/niklhart/kldest/issues

NeedsCompilation no

Author Niklas Hartung [aut, cre, cph]
(<https://orcid.org/0000-0002-4000-6525>)

Repository CRAN

Date/Publication 2024-04-09 08:20:02 UTC

R topics documented:

`combinations` *Combinations of input arguments*

Description

Combinations of input arguments

Usage

```
combinations(...)
```

Arguments

`...`	Any number of atomic vectors.

Value

A data frame with columns named as the inputs, containing all input combinations.

Examples

```
combinations(a = 1:2, b = letters[1:3], c = LETTERS[1:2])
```

constDiagMatrix *Constant plus diagonal matrix*

Description

Specify a matrix with constant values on the diagonal and on the off-diagonals. Such matrices can be used to vary the degree of dependency in covariate matrices, for example when evaluating accuracy of KL-divergence estimation algorithms.

Usage

```
constDiagMatrix(dim = 1, diag = 1, offDiag = 0)
```

Arguments

`dim`	Dimension
`diag`	Value at the diagonal
`offDiag`	Value at off-diagonals

Value

A `dim`-by-`dim` matrix

Examples

```
constDiagMatrix(dim = 3, diag = 1, offDiag = 0.9)
```

convergence_rate *Empirical convergence rate of a KL divergence estimator*

Description

Subsampling-based confidence intervals computed by `kld_ci_subsampling()` require the convergence rate of the KL divergence estimator as an input. The default rate of `0.5` assumes that the variance term dominates the bias term. For high-dimensional problems, depending on the data, the convergence rate might be lower. This function allows to empirically derive the convergence rate.

Usage

```
convergence_rate(
  estimator,
  X,
  Y = NULL,
  q = NULL,
  n.sizes = 4,
  spacing.factor = 1.5,
  typical.subsample = function(n) sqrt(n),
  B = 500L,
  plot = FALSE
)
```

Arguments

`estimator`	A KL divergence estimator.
`X, Y`	`n`-by-`d` and `m`-by-`d` data frames or matrices (multivariate samples), or numeric/character vectors (univariate samples, i.e. `d = 1`), representing `n` samples from the true distribution P and `m` samples from the approximate distribution Q in `d` dimensions. `Y` can be left blank if `q` is specified (see below).
`q`	The density function of the approximate distribution Q. Either `Y` or `q` must be specified. If the distributions are all continuous or all discrete, `q` can be directly specified as the probability density/mass function. However, for mixed continuous/discrete distributions, `q` must be given in decomposed form, $q(y_c, y_d) = q_{c\|d}(y_c\|y_d)q_d(y_d)$, specified as a named list with field `cond` for the conditional density $q_{c\|d}(y_c\|y_d)$ (a function that expects two arguments `y_c` and `y_d`) and `disc` for the discrete marginal density $q_d(y_d)$ (a function that expects one argument `y_d`). If such a decomposition is not available, it may be preferable to instead simulate a large sample from Q and use the two-sample syntax.
`n.sizes`	Number of different subsample sizes to use (default: `4`).
`spacing.factor`	Multiplicative factor controlling the spacing of sample sizes (default: `1.5`).
`typical.subsample`	A function that produces a typical subsample size, used as the geometric mean of subsample sizes (default: `sqrt(n)`).
`B`	Number of subsamples to draw per subsample size.
`plot`	A boolean (default: `FALSE`) controlling whether to produce a diagnostic plot visualizing the fit.

Details

References:

Politis, Romano and Wolf, "Subsampling", Chapter 8 (1999), for theory.

The implementation has been adapted from lecture notes by C. J. Geyer, https://www.stat.umn.edu/geyer/5601/notes/sub.pdf

Value

A scalar, the parameter β in the empirical convergence rate $n^-\beta$ of the `estimator` to the true KL divergence. It can be used in the `convergence.rate` argument of `kld_ci_subsampling()` as `convergence.rate = function(n) n^beta`.

Examples

```
# NN method usually has a convergence rate around 0.5:
set.seed(0)
convergence_rate(kld_est_nn, X = rnorm(1000), Y = rnorm(1000, mean = 1, sd = 2))
```

`is_two_sample` *Detect if a one- or two-sample problem is specified*

Description

Detect if a one- or two-sample problem is specified

Usage

```
is_two_sample(Y, q)
```

Arguments

`Y`	A vector, matrix, data frame or `NULL`
`q`	A function or `NULL`.

Value

`TRUE` for a two-sample problem (i.e., `Y` non-null and `q = NULL`) and `FALSE` for a one-sample problem (i.e., `Y = NULL` and `q` non-null).

`kld_ci_bootstrap` *Uncertainty of KL divergence estimate using Efron's bootstrap.*

Description

This function computes a confidence interval for KL divergence based on Efron's bootstrap. The approach only works for kernel density-based estimators since nearest neighbour-based estimators cannot deal with the ties produced when sampling with replacement.

Usage

```
kld_ci_bootstrap(
  X,
  Y,
  estimator = kld_est_kde1,
  B = 500L,
  alpha = 0.05,
  method = c("quantile", "se"),
  include.boot = FALSE,
  ...
)
```

Arguments

X, Y	n-by-d and m-by-d matrices, representing n samples from the true distribution P and m samples from the approximate distribution Q, both in d dimensions. Vector input is treated as a column matrix.
estimator	A function expecting two inputs X and Y, the Kullback-Leibler divergence estimation method. Defaults to kld_est_kde1, which can only deal with one-dimensional two-sample problems (i.e., d = 1 and q = NULL).
B	Number of bootstrap replicates (default: 500), the larger, the more accurate, but also more computationally expensive.
alpha	Error level, defaults to 0.05.
method	Either "quantile" (the default), also known as the reverse percentile method, or "se" for a normal approximation of the KL divergence estimator using the standard error of the subsamples.
include.boot	Boolean, TRUE means KL divergene estimates on bootstrap samples are included in the returned list.
...	Arguments passed on to estimator, i.e. as estimator(X, Y, ...).

Details

Reference:

Efron, "Bootstrap Methods: Another Look at the Jackknife", The Annals of Statistics, Vol. 7, No. 1 (1979).

Value

A list with the following fields:

- "est" (the estimated KL divergence),
- "boot" (a length B numeric vector with KL divergence estimates on the bootstrap subsamples), only included if include.boot = TRUE,
- "ci" (a length 2 vector containing the lower and upper limits of the estimated confidence interval).

Examples

```
# 1D Gaussian, two samples
set.seed(0)
X <- rnorm(100)
Y <- rnorm(100, mean = 1, sd = 2)
kld_gaussian(mu1 = 0, sigma1 = 1, mu2 = 1, sigma2 = 2^2)
kld_est_kde1(X, Y)
kld_ci_bootstrap(X, Y)
```

`kld_ci_subsampling` *Uncertainty of KL divergence estimate using Politis/Romano's subsampling bootstrap.*

Description

This function computes a confidence interval for KL divergence based on the subsampling bootstrap introduced by Politis and Romano. See **Details** for theoretical properties of this method.

Usage

```
kld_ci_subsampling(
  X,
  Y = NULL,
  q = NULL,
  estimator = kld_est_nn,
  B = 500L,
  alpha = 0.05,
  subsample.size = function(x) x^(2/3),
  convergence.rate = sqrt,
  method = c("quantile", "se"),
  include.boot = FALSE,
  n.cores = 1L,
  ...
)
```

Arguments

`X, Y` — `n`-by-`d` and `m`-by-`d` data frames or matrices (multivariate samples), or numeric/character vectors (univariate samples, i.e. `d = 1`), representing `n` samples from the true distribution P and `m` samples from the approximate distribution Q in `d` dimensions. `Y` can be left blank if `q` is specified (see below).

`q` — The density function of the approximate distribution Q. Either `Y` or `q` must be specified. If the distributions are all continuous or all discrete, `q` can be directly specified as the probability density/mass function. However, for mixed continuous/discrete distributions, `q` must be given in decomposed form, $q(y_c, y_d) = q_{c|d}(y_c|y_d)q_d(y_d)$, specified as a named list with field `cond` for the conditional density $q_{c|d}(y_c|y_d)$ (a function that expects two arguments `y_c` and `y_d`) and `disc` for the discrete marginal density $q_d(y_d)$ (a function that expects one argument `y_d`). If such a decomposition is not available, it may be preferable to instead simulate a large sample from Q and use the two-sample syntax.

`estimator` The Kullback-Leibler divergence estimation method; a function expecting two inputs (`X` and `Y` or `q`, depending on arguments provided). Defaults to `kld_est_nn`.

`B` Number of bootstrap replicates (default: `500`), the larger, the more accurate, but also more computationally expensive.

`alpha` Error level, defaults to `0.05`.

`subsample.size` A function specifying the size of the subsamples, defaults to $f(x) = x^{2/3}$.

`convergence.rate` A function computing the convergence rate of the estimator as a function of sample sizes. Defaults to $f(x) = x^{1/2}$. If `convergence.rate` is `NULL`, it is estimated empirically from the sample(s) using `kldest::convergence_rate()`.

`method` Either `"quantile"` (the default), also known as the reverse percentile method, or `"se"` for a normal approximation of the KL divergence estimator using the standard error of the subsamples.

`include.boot` Boolean, `TRUE` means KL divergence estimates on subsamples are included in the returned list. Defaults to `FALSE`.

`n.cores` Number of cores to use in parallel computing (defaults to `1`, which means that no parallel computing is used). To use this option, the `parallel` package must be installed and the OS must be of UNIX type (i.e., not Windows). Otherwise, `n.cores` will be reset to `1`, with a message.

`...` Arguments passed on to `estimator`, i.e. via the call `estimator(X, Y = Y, ...)` or `estimator(X, q = q, ...)`.

Details

In general terms, tetting b_n be the subsample size for a sample of size n, and τ_n the convergence rate of the estimator, a confidence interval calculated by subsampling has asymptotic coverage $1-\alpha$ as long as $b_n/n \to 0$, $b_n \to \infty$ and $\frac{\tau_{b_n}}{\tau_n} \to 0$.

In many cases, the convergence rate of the nearest-neighbour based KL divergence estimator is $\tau_n = \sqrt{n}$ and the condition on the subsample size reduces to $b_n/n \to 0$ and $b_n \to \infty$. By default, $b_n = n^{2/3}$. In a two-sample problem, n and b_n are replaced by effective sample sizes $n_{\text{eff}} = \min(n, m)$ and $b_{n,\text{eff}} = \min(b_n, b_m)$.

Reference:

Politis and Romano, "Large sample confidence regions based on subsamples under minimal assumptions", The Annals of Statistics, Vol. 22, No. 4 (1994).

Value

A list with the following fields:

- `"est"` (the estimated KL divergence),
- `"ci"` (a length `2` vector containing the lower and upper limits of the estimated confidence interval).

- "boot" (a length B numeric vector with KL divergence estimates on the bootstrap subsamples), only included if include.boot = TRUE,

Examples

```
# 1D Gaussian (one- and two-sample problems)
set.seed(0)
X <- rnorm(100)
Y <- rnorm(100, mean = 1, sd = 2)
q <- function(x) dnorm(x, mean =1, sd = 2)
kld_gaussian(mu1 = 0, sigma1 = 1, mu2 = 1, sigma2 = 2^2)
kld_est_nn(X, Y = Y)
kld_est_nn(X, q = q)
kld_ci_subsampling(X, Y)$ci
kld_ci_subsampling(X, q = q)$ci
```

---

kld_discrete *Analytical KL divergence for two discrete distributions*

Description

Analytical KL divergence for two discrete distributions

Usage

```
kld_discrete(P, Q)
```

Arguments

P, Q	Numerical arrays with the same dimensions, representing discrete probability distributions

Value

A scalar (the Kullback-Leibler divergence)

Examples

```
# 1-D example
P <- 1:4/10
Q <- rep(0.25,4)
kld_discrete(P,Q)

# The above example in 2-D
P <- matrix(1:4/10,nrow=2)
Q <- matrix(0.25,nrow=2,ncol=2)
kld_discrete(P,Q)
```

`kld_est` *Kullback-Leibler divergence estimator for discrete, continuous or mixed data.*

Description

For two mixed continuous/discrete distributions with densities p and q, and denoting $x = (x_\mathrm{c}, x_\mathrm{d})$, the Kullback-Leibler divergence $D_{KL}(p||q)$ is given as

$$D_{KL}(p||q) = \sum_{x_d} \int p(x_c, x_d) \log\left(\frac{p(x_c, x_d)}{q(x_c, x_d)}\right) dx_c.$$

Conditioning on the discrete variables x_d, this can be re-written as

$$D_{KL}(p||q) = \sum_{x_d} p(x_d) D_{KL}\big(p(\cdot|x_d)||q(\cdot|x_d)\big) + D_{KL}\big(p_{x_d}||q_{x_d}\big).$$

Here, the terms

$$D_{KL}\big(p(\cdot|x_d)||q(\cdot|x_d)\big)$$

are approximated via nearest neighbour- or kernel-based density estimates on the datasets `X` and `Y` stratified by the discrete variables, and

$$D_{KL}\big(p_{x_d}||q_{x_d}\big)$$

is approximated using relative frequencies.

Usage

```
kld_est(
  X,
  Y = NULL,
  q = NULL,
  estimator.continuous = kld_est_nn,
  estimator.discrete = kld_est_discrete,
  vartype = NULL
)
```

Arguments

`X, Y` n-by-d and m-by-d data frames or matrices (multivariate samples), or numeric/character vectors (univariate samples, i.e. `d = 1`), representing `n` samples from the true distribution P and `m` samples from the approximate distribution Q in `d` dimensions. `Y` can be left blank if `q` is specified (see below).

`q` The density function of the approximate distribution Q. Either `Y` or `q` must be specified. If the distributions are all continuous or all discrete, `q` can be directly specified as the probability density/mass function. However, for mixed continuous/discrete distributions, `q` must be given in decomposed form, $q(y_c, y_d) =$

$q_{c|d}(y_c|y_d)q_d(y_d)$, specified as a named list with field cond for the conditional density $q_{c|d}(y_c|y_d)$ (a function that expects two arguments y_c and y_d) and disc for the discrete marginal density $q_d(y_d)$ (a function that expects one argument y_d). If such a decomposition is not available, it may be preferable to instead simulate a large sample from Q and use the two-sample syntax.

estimator.continuous, estimator.discrete
: KL divergence estimators for continuous and discrete data, respectively. Both are functions with two arguments X and Y or X and q, depending on whether a two-sample or one-sample problem is considered. Defaults are kld_est_nn and kld_est_discrete, respectively.

vartype
: A length d character vector, with vartype[i] = "c" meaning the i-th variable is continuous, and vartype[i] = "d" meaning it is discrete. If unspecified, vartype is "c" for numeric columns and "d" for character or factor columns. This default will mostly work, except if levels of discrete variables are encoded using numbers (e.g., 0 for females and 1 for males) or for count data.

Value

A scalar, the estimated Kullback-Leibler divergence $\hat{D}_{KL}(P||Q)$.

Examples

```
# 2D example, two samples
set.seed(0)
X <- data.frame(cont  = rnorm(10),
                discr = c(rep('a',4),rep('b',6)))
Y <- data.frame(cont  = c(rnorm(5), rnorm(5, sd = 2)),
                discr = c(rep('a',5),rep('b',5)))
kld_est(X, Y)

# 2D example, one sample
set.seed(0)
X <- data.frame(cont  = rnorm(10),
                discr = c(rep(0,4),rep(1,6)))
q <- list(cond = function(xc,xd) dnorm(xc, mean = xd, sd = 1),
          disc = function(xd) dbinom(xd, size = 1, prob = 0.5))
kld_est(X, q = q, vartype = c("c","d"))
```

---

kld_est_brnn — *Bias-reduced generalized k-nearest-neighbour KL divergence estimation*

Description

This is the bias-reduced generalized k-NN based KL divergence estimator from Wang et al. (2009) specified in Eq.(29).

Usage

```
kld_est_brnn(X, Y, max.k = 100, warn.max.k = TRUE, eps = 0)
```

Arguments

X, Y	n-by-d and m-by-d matrices, representing n samples from the true distribution P and m samples from the approximate distribution Q, both in d dimensions. Vector input is treated as a column matrix. Y can be left blank if q is specified (see below).
max.k	Maximum numbers of nearest neighbours to compute (default: 100). A larger max.k may yield a more accurate KL-D estimate (see warn.max.k), but will always increase the computational cost.
warn.max.k	If TRUE (the default), warns if max.k is such that more than max.k neighbours are within the neighbourhood δ for some data point(s). In this case, only the first max.k neighbours are counted. As a consequence, max.k may required to be increased.
eps	Error bound in the nearest neighbour search. A value of eps = 0 (the default) implies an exact nearest neighbour search, for eps > 0 approximate nearest neighbours are sought, which may be somewhat faster for high-dimensional problems.

Details

Finite sample bias reduction is achieved by an adaptive choice of the number of nearest neighbours. Fixing the number of nearest neighbours upfront, as done in kld_est_nn(), may result in very different distances ρ^l_i, ν^k_i of a datapoint x_i to its l-th nearest neighbours in X and k-th nearest neighbours in Y, respectively, which may lead to unequal biases in NN density estimation, especially in a high-dimensional setting. To overcome this issue, the number of neighbours l, k are here chosen in a way to render ρ^l_i, ν^k_i comparable, by taking the largest possible number of neighbours l_i, k_i smaller than $\delta_i := \max(\rho^1_i, \nu^1_i)$.

Since the bias reduction explicitly uses both samples X and Y, one-sample estimation is not possible using this method.

Reference: Wang, Kulkarni and Verdú, "Divergence Estimation for Multidimensional Densities Via k-Nearest-Neighbor Distances", IEEE Transactions on Information Theory, Vol. 55, No. 5 (2009). DOI: https://doi.org/10.1109/TIT.2009.2016060

Value

A scalar, the estimated Kullback-Leibler divergence $\hat{D}_{KL}(P||Q)$.

Examples

```
# KL-D between one or two samples from 1-D Gaussians:
set.seed(0)
X <- rnorm(100)
Y <- rnorm(100, mean = 1, sd = 2)
q <- function(x) dnorm(x, mean = 1, sd =2)
kld_gaussian(mu1 = 0, sigma1 = 1, mu2 = 1, sigma2 = 2^2)
kld_est_nn(X, Y)
```

```
kld_est_nn(X, q = q)
kld_est_nn(X, Y, k = 5)
kld_est_nn(X, q = q, k = 5)
kld_est_brnn(X, Y)


# KL-D between two samples from 2-D Gaussians:
set.seed(0)
X1 <- rnorm(100)
X2 <- rnorm(100)
Y1 <- rnorm(100)
Y2 <- Y1 + rnorm(100)
X <- cbind(X1,X2)
Y <- cbind(Y1,Y2)
kld_gaussian(mu1 = rep(0,2), sigma1 = diag(2),
             mu2 = rep(0,2), sigma2 = matrix(c(1,1,1,2),nrow=2))
kld_est_nn(X, Y)
kld_est_nn(X, Y, k = 5)
kld_est_brnn(X, Y)
```

`kld_est_discrete` *Plug-in KL divergence estimator for samples from discrete distributions*

Description

Plug-in KL divergence estimator for samples from discrete distributions

Usage

```
kld_est_discrete(X, Y = NULL, q = NULL)
```

Arguments

`X, Y`	`n`-by-`d` and `m`-by-`d` matrices or data frames, representing `n` samples from the true discrete distribution P and `m` samples from the approximate discrete distribution Q, both in `d` dimensions. Vector input is treated as a column matrix. Argument `Y` can be omitted if argument `q` is given (see below).
`q`	The probability mass function of the approximate distribution Q. Currently, the one-sample problem is only implemented for `d=1`.

Value

A scalar, the estimated Kullback-Leibler divergence $\hat{D}_{KL}(P||Q)$.

Examples

```
# 1D example, two samples
X <- c(rep('M',5),rep('F',5))
Y <- c(rep('M',6),rep('F',4))
kld_est_discrete(X, Y)

# 1D example, one sample
X <- c(rep(0,4),rep(1,6))
q <- function(x) dbinom(x, size = 1, prob = 0.5)
kld_est_discrete(X, q = q)
```

`kld_est_kde` *Kernel density-based Kullback-Leibler divergence estimation in any dimension*

Description

Disclaimer: this function doesn't use binning and/or the fast Fourier transform and hence, it is extremely slow even for moderate datasets. For this reason, it is not exported currently.

Usage

```
kld_est_kde(X, Y, hX = NULL, hY = NULL, rule = c("Silverman", "Scott"))
```

Arguments

`X, Y`	`n`-by-`d` and `m`-by-`d` matrices, representing `n` samples from the true distribution P and `m` samples from the approximate distribution Q, both in `d` dimensions. Vector input is treated as a column matrix.
`hX, hY`	Positive scalars or length `d` vectors, representing bandwidth parameters (possibly different in each component) for the density estimates of P and Q, respectively. If unspecified, a heurestic specified via the `rule` argument is used.
`rule`	A heuristic for computing arguments `hX` and/or `hY`. The default `"silverman"` is Silverman's rule $$h_i = \sigma_i \left(\frac{4}{(2+d)n}\right)^{1/(d+4)}.$$ As an alternative, Scott's rule `"scott"` can be used, $$h_i = \frac{\sigma_i}{n^{1/(d+4)}}.$$

Details

This estimation method approximates the densities of the unknown distributions P and Q by kernel density estimates, using a sample size- and dimension-dependent bandwidth parameter and a Gaussian kernel. It works for any number of dimensions but is very slow.

Value

A scalar, the estimated Kullback-Leibler divergence $\hat{D}_{KL}(P||Q)$.

Examples

```
# KL-D between two samples from 1-D Gaussians:
set.seed(0)
X <- rnorm(100)
Y <- rnorm(100, mean = 1, sd = 2)
kld_gaussian(mu1 = 0, sigma1 = 1, mu2 = 1, sigma2 = 2^2)
kld_est_kde1(X, Y)
kld_est_nn(X, Y)
kld_est_brnn(X, Y)

# KL-D between two samples from 2-D Gaussians:
set.seed(0)
X1 <- rnorm(100)
X2 <- rnorm(100)
Y1 <- rnorm(100)
Y2 <- Y1 + rnorm(100)
X <- cbind(X1,X2)
Y <- cbind(Y1,Y2)
kld_gaussian(mu1 = rep(0,2), sigma1 = diag(2),
             mu2 = rep(0,2), sigma2 = matrix(c(1,1,1,2),nrow=2))
kld_est_kde2(X, Y)
kld_est_nn(X, Y)
kld_est_brnn(X, Y)
```

---

kld_est_kde1 — *1-D kernel density-based estimation of Kullback-Leibler divergence*

Description

This estimation method approximates the densities of the unknown distributions P and Q by a kernel density estimate using function 'density' from package 'stats'. Only the two-sample, not the one-sample problem is implemented.

Usage

```
kld_est_kde1(X, Y, MC = FALSE, ...)
```

Arguments

`X, Y`	Numeric vectors or single-column matrices, representing samples from the true distribution P and the approximate distribution Q, respectively.
`MC`	A boolean: use a Monte Carlo approximation instead of numerical integration via the trapezoidal rule (default: `FALSE`)?
`...`	Further parameters to passed on to `stats::density` (e.g., argument `bw`)

Value

A scalar, the estimated Kullback-Leibler divergence $\hat{D}_{KL}(P||Q)$.

Examples

```
# KL-D between two samples from 1D Gaussians:
set.seed(0)
X <- rnorm(100)
Y <- rnorm(100, mean = 1, sd = 2)
kld_gaussian(mu1 = 0, sigma1 = 1, mu2 = 1, sigma2 = 2^2)
kld_est_kde1(X,Y)
kld_est_kde1(X,Y, MC = TRUE)
```

`kld_est_kde2` *2-D kernel density-based estimation of Kullback-Leibler divergence*

Description

This estimation method approximates the densities of the unknown bivariate distributions P and Q by kernel density estimates using function 'bkde' from package 'KernSmooth'. If 'KernSmooth' is not installed, a message is issued and the (much) slower function 'kld_est_kde' is used instead.

Usage

```
kld_est_kde2(
  X,
  Y,
  MC = FALSE,
  hX = NULL,
  hY = NULL,
  rule = c("Silverman", "Scott"),
  eps = 1e-05
)
```

Arguments

`X, Y`	`n`-by-`2` and `m`-by-`2` matrices, representing `n` samples from the bivariate true distribution P and `m` samples from the approximate distribution Q, respectively.
`MC`	A boolean: use a Monte Carlo approximation instead of numerical integration via the trapezoidal rule (default: `FALSE`)? Currently, this option is not implemented, i.e. a value of `TRUE` results in an error.
`hX, hY`	Bandwidths for the kernel density estimates of P and Q, respectively. The default `NULL` means they are determined by argument `rule`.
`rule`	A heuristic to derive parameters `hX` and `hY`, default is `"Silverman", which means that`

$$h_i = \sigma_i \left(\frac{4}{(2+d)n} \right)^{1/(d+4)}.$$

`eps`	A nonnegative scalar; if `eps > 0`, Q is estimated as a mixture between the kernel density estimate and a uniform distribution on the computational grid. The weight of the uniform component is `eps` times the maximum density estimate of Q. This increases the robustness of the estimator at the expense of an additional bias. Defaults to `eps = 1e-5`.

Value

A scalar, the estimated Kullback-Leibler divergence $\hat{D}_{KL}(P||Q)$.

Examples

```
# KL-D between two samples from 2-D Gaussians:
set.seed(0)
X1 <- rnorm(1000)
X2 <- rnorm(1000)
Y1 <- rnorm(1000)
Y2 <- Y1 + rnorm(1000)
X <- cbind(X1,X2)
Y <- cbind(Y1,Y2)
kld_gaussian(mu1 = rep(0,2), sigma1 = diag(2),
             mu2 = rep(0,2), sigma2 = matrix(c(1,1,1,2),nrow=2))
kld_est_kde2(X,Y)
```

`kld_est_nn` *k-nearest neighbour KL divergence estimator*

Description

This function estimates Kullback-Leibler divergence $D_{KL}(P||Q)$ between two continuous distributions P and Q using nearest-neighbour (NN) density estimation in a Monte Carlo approximation of $D_{KL}(P||Q)$.

Usage

```
kld_est_nn(X, Y = NULL, q = NULL, k = 1L, eps = 0, log.q = FALSE)
```

Arguments

`X, Y`	`n`-by-`d` and `m`-by-`d` matrices, representing `n` samples from the true distribution P and `m` samples from the approximate distribution Q, both in `d` dimensions. Vector input is treated as a column matrix. `Y` can be left blank if `q` is specified (see below).
`q`	The density function of the approximate distribution Q. Either `Y` or `q` must be specified.
`k`	The number of nearest neighbours to consider for NN density estimation. Larger values for `k` generally increase bias, but decrease variance of the estimator. Defaults to `k = 1`.

eps	Error bound in the nearest neighbour search. A value of eps = 0 (the default) implies an exact nearest neighbour search, for eps > 0 approximate nearest neighbours are sought, which may be somewhat faster for high-dimensional problems.
log.q	If TRUE, function q is the log-density rather than the density of the approximate distribution Q (default: log.q = FALSE).

Details

Input for estimation is a sample X from P and either the density function q of Q (one-sample problem) or a sample Y of Q (two-sample problem). In the two-sample problem, it is the estimator in Eq.(5) of Wang et al. (2009). In the one-sample problem, the asymptotic bias (the expectation of a Gamma distribution) is substracted, see Pérez-Cruz (2008), Eq.(18).

References:

Wang, Kulkarni and Verdú, "Divergence Estimation for Multidimensional Densities Via k-Nearest-Neighbor Distances", IEEE Transactions on Information Theory, Vol. 55, No. 5 (2009).

Pérez-Cruz, "Kullback-Leibler Divergence Estimation of Continuous Distributions", IEEE International Symposium on Information Theory (2008).

Value

A scalar, the estimated Kullback-Leibler divergence $\hat{D}_{KL}(P||Q)$.

Examples

```
# KL-D between one or two samples from 1-D Gaussians:
set.seed(0)
X <- rnorm(100)
Y <- rnorm(100, mean = 1, sd = 2)
q <- function(x) dnorm(x, mean = 1, sd =2)
kld_gaussian(mu1 = 0, sigma1 = 1, mu2 = 1, sigma2 = 2^2)
kld_est_nn(X, Y)
kld_est_nn(X, q = q)
kld_est_nn(X, Y, k = 5)
kld_est_nn(X, q = q, k = 5)
kld_est_brnn(X, Y)


# KL-D between two samples from 2-D Gaussians:
set.seed(0)
X1 <- rnorm(100)
X2 <- rnorm(100)
Y1 <- rnorm(100)
Y2 <- Y1 + rnorm(100)
X <- cbind(X1,X2)
Y <- cbind(Y1,Y2)
kld_gaussian(mu1 = rep(0,2), sigma1 = diag(2),
             mu2 = rep(0,2), sigma2 = matrix(c(1,1,1,2),nrow=2))
kld_est_nn(X, Y)
kld_est_nn(X, Y, k = 5)
kld_est_brnn(X, Y)
```

kld_exponential — *Analytical KL divergence for two univariate exponential distributions*

Description

This function computes $D_{KL}(p||q)$, where $p \sim \text{Exp}(\lambda_1)$ and $q \sim \text{Exp}(\lambda_2)$, in rate parametrization.

Usage

```
kld_exponential(lambda1, lambda2)
```

Arguments

`lambda1`	A scalar (rate parameter of true exponential distribution)
`lambda2`	A scalar (rate parameter of approximate exponential distribution)

Value

A scalar (the Kullback-Leibler divergence)

Examples

```
kld_exponential(lambda1 = 1, lambda2 = 2)
```

kld_gaussian — *Analytical KL divergence for two uni- or multivariate Gaussian distributions*

Description

This function computes $D_{KL}(p||q)$, where $p \sim \mathcal{N}(\mu_1, \Sigma_1)$ and $q \sim \mathcal{N}(\mu_2, \Sigma_2)$.

Usage

```
kld_gaussian(mu1, sigma1, mu2, sigma2)
```

Arguments

`mu1`	A numeric vector (mean of true Gaussian)
`sigma1`	A s.p.d. matrix (Covariance matrix of true Gaussian)
`mu2`	A numeric vector (mean of approximate Gaussian)
`sigma2`	A s.p.d. matrix (Covariance matrix of approximate Gaussian)

Value

A scalar (the Kullback-Leibler divergence)

Examples

```
kld_gaussian(mu1 = 1, sigma1 = 1, mu2 = 1, sigma2 = 2^2)
kld_gaussian(mu1 = rep(0,2), sigma1 = diag(2),
                mu2 = rep(1,2), sigma2 = matrix(c(1,0.5,0.5,1), nrow = 2))
```

---

kld_uniform *Analytical KL divergence for two uniform distributions*

Description

This function computes $D_{KL}(p||q)$, where $p \sim \mathrm{U}(a_1, b_1)$ and $q \sim \mathrm{U}(a_2, b_2)$, with $a_2 < a_1 < b_1 < b_2$.

Usage

```
kld_uniform(a1, b1, a2, b2)
```

Arguments

a1, b1	Range of true uniform distribution
a2, b2	Range of approximate uniform distribution

Value

A scalar (the Kullback-Leibler divergence)

Examples

```
kld_uniform(a1 = 0, b1 = 1, a2 = 0, b2 = 2)
```

---

kld_uniform_gaussian *Analytical KL divergence between a uniform and a Gaussian distribution*

Description

This function computes $D_{KL}(p||q)$, where $p \sim \mathrm{U}(a, b)$ and $q \sim \mathcal{N}(\mu, \sigma^2)$.

Usage

```
kld_uniform_gaussian(a = 0, b = 1, mu = 0, sigma2 = 1)
```

Arguments

a, b	Parameters of uniform (true) distribution
mu, sigma2	Parameters of Gaussian (approximate) distribution

Value

A scalar (the Kullback-Leibler divergence)

Examples

```
kld_uniform_gaussian(a = 0, b = 1, mu = 0, sigma2 = 1)
```

---

mvdnorm *Probability density function of multivariate Gaussian distribution*

Description

Probability density function of multivariate Gaussian distribution

Usage

```
mvdnorm(x, mu, Sigma)
```

Arguments

x	A vector of length d at which Gaussian density is evaluated.
mu	A vector of length d, mean of Gaussian distribution.
Sigma	A d-by-d matrix, covariance matrix of Gaussian distribution.

Value

The probability density of $N(\mu, \Sigma)$ evaluated at x.

Examples

```
# 1D example
mvdnorm(x = 2, mu = 1, Sigma = 2)
dnorm(x = 2, mean = 1, sd = sqrt(2))
# Independent 2D example
mvdnorm(x = c(2,2), mu = c(1,1), Sigma = diag(1:2))
prod(dnorm(x = c(2,2), mean = c(1,1), sd = sqrt(1:2)))
# Correlated 2D example
mvdnorm(x = c(2,2), mu = c(1,1), Sigma = matrix(c(2,1,1,2),nrow=2))
```

to_uniform_scale *Transform samples to uniform scale*

Description

Since Kullback-Leibler divergence is scale-invariant, its sample-based approximations can be computed on a conveniently chosen scale. This helper functions transforms each variable in a way that all marginal distributions of the joint dataset (X, Y) are uniform. In this way, the scales of different variables are rendered comparable, with the idea of a better performance of neighbour-based methods in this situation.

Usage

```
to_uniform_scale(X, Y)
```

Arguments

X, Y	n-by-d and m-by-d matrices, representing n samples from the true distribution P and m samples from the approximate distribution Q, both in d dimensions. Vector input is treated as a column matrix. Y can be left blank if q is specified (see below).

Value

A list with fields X and Y, containing the transformed samples.

Examples

```
# 2D example
n <- 10L
X <- cbind(rnorm(n, mean = 0, sd = 3),
           rnorm(n, mean = 1, sd = 2))
Y <- cbind(rnorm(n, mean = 1, sd = 2),
           rnorm(n, mean = 0, sd = 2))
to_uniform_scale(X, Y)
```

tr *Matrix trace operator*

Description

Matrix trace operator

Usage

```
tr(M)
```

Arguments

M	A square matrix

Value

The matrix trace (a scalar)

`trapz` *Trapezoidal integration in 1 or 2 dimensions*

Description

Trapezoidal integration in 1 or 2 dimensions

Usage

```
trapz(h, fx)
```

Arguments

h	A length d numeric vector of grid widths.
fx	A d-dimensional array (or a vector, if d=1).

Value

The trapezoidal approximation of the integral.

Examples

```
# 1D example
trapz(h = 1, fx = 1:10)
# 2D example
trapz(h = c(1,1), fx = matrix(1:10, nrow = 2))
```

Index