Package 'RobLoxBioC'

January 20, 2025

Version 1.2.3

Date 2025-01-19

Title Infinitesimally Robust Estimators for Preprocessing -Omics Data

Description Functions for the determination of optimally robust influence curves and estimators for preprocessing omics data, in particular gene expression data (Kohl and Deigner (2010), <doi:10.1186/1471-2105-11-583>).

Depends R(>= 3.4), methods, distr(>= 2.8.0), affy

Imports Biobase, BiocGenerics, beadarray, RobLox(>= 1.1.0), distrMod(>= 2.8.0), lattice, RColorBrewer, AnnotationDbi

Suggests affydata, hgu95av2cdf, beadarrayExampleData, illuminaHumanv3.db

ByteCompile yes

License LGPL-3

URL https://r-forge.r-project.org/projects/robast/

LastChangedDate {$LastChangedDate: 2025-01-19 17:58:14 +0100 (So, 19. Jan 2025) $}

LastChangedRevision {$LastChangedRevision: 1333 $}

VCS/SVNRevision 1329

NeedsCompilation no

Author Matthias Kohl [aut, cre, cph]

Maintainer Matthias Kohl <Matthias.Kohl@stamats.de>

Repository CRAN

Date/Publication 2025-01-19 17:20:02 UTC

Contents

RobLoxBioC-package *Infinitesimally robust estimators for preprocessing omics data*

Description

Functions for the determination of optimally robust influence curves and estimators for preprocessing omics data, in particular gene expression data (Kohl and Deigner (2010)).

Package versions

Note: The first two numbers of package versions do not necessarily reflect package-individual development, but rather are chosen for the RobAStXXX family as a whole in order to ease updating "depends" information.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>
Maintainer: Matthias Kohl <matthias.kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

Kohl M. and Deigner H.P. (2010). Preprocessing of gene expression data by optimally robust estimators. *BMC Bioinformatics*, 11:583.

M. Kohl, P. Ruckdeschel, and H. Rieder (2010). Infinitesimally Robust Estimation in General Smoothly Parametrized Models. *Statistical Methods and Application*, **19**(3):333-354.

Rieder, H. (1994) *Robust Asymptotic Statistics*. New York: Springer.

Rieder, H., Kohl, M. and Ruckdeschel, P. (2008) The Costs of not Knowing the Radius. *Statistical Methods and Applications* **17**(1) 13-40.

See Also

roblox, rowRoblox

Examples

```
library(RobLoxBioC)
```

KolmogorovMinDist *Generic Function for Computing Minimum Kolmogorov Distance for Biological Data*

Description

Generic function for computing minimum Kolmogorov distance for biological data.

Usage

```
KolmogorovMinDist(x, D, ...)

## S4 method for signature 'matrix,Norm'
KolmogorovMinDist(x, D, mad0 = 1e-4)

## S4 method for signature 'AffyBatch,AbscontDistribution'
KolmogorovMinDist(x, D, bg.correct = TRUE, pmcorrect = TRUE,
                  verbose = TRUE)

## S4 method for signature 'beadLevelData,AbscontDistribution'
KolmogorovMinDist(x, D, log = FALSE, what = "Grn",
                  probes = NULL, arrays = NULL)
```

Arguments

x	biological data.
D	object of class "UnivariateDistribution".
...	additional parameters.
mad0	scale estimate used if computed MAD is equal to zero. Median and MAD are used as start parameter for optimization.
bg.correct	if TRUE MAS 5.0 background correction is performed; confer bg.correct.mas.
pmcorrect	if TRUE log2(PM/MM) is used. If FALSE only log2(PM) is used.
verbose	logical: if TRUE, some messages are printed.
log	if TRUE, then the log2 intensities for each bead-type are summarized.
what	character string specifying which intensities/values to summarize; see getBeadData.
probes	Specify particular probes to summarize. If left NULL then all the probes on the first array are used.
arrays	integer (scalar or vector) specifying the strips/arrays to summarize. If NULL, then all strips/arrays are summarized.

Details

The minimum Kolmogorov distance is computed for each row of a matrix, each Affymetrix probe, or each Illumina bead, respectively.

So far, only the minimum distance to the set of normal distributions can be computed.

Value

List with components `dist` containing a numeric vector or matrix with minimum Kolmogorov distances and `n` a numeric vector or matrix with the corresponding sample sizes.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1981) *Robust Statistics*. New York: Wiley.

Rieder, H. (1994) *Robust Asymptotic Statistics*. New York: Springer.

See Also

KolmogorovDist, MDEstimator

Examples

```
set.seed(123) # to have reproducible results for package checking

## matrix method for KolmogorovMinDist
ind <- rbinom(200, size=1, prob=0.05)
X <- matrix(rnorm(200, mean=ind*3, sd=(1-ind) + ind*9), nrow = 2)
KolmogorovMinDist(X, D = Norm())

## using Affymetrix data
data(SpikeIn)
probes <- log2(pm(SpikeIn))
(res <- KolmogorovMinDist(probes, Norm()))
boxplot(res$dist)


## \donttest because of check time
## using Affymetrix data
library(affydata)
data(Dilution)
res <- KolmogorovMinDist(Dilution[,1], Norm())
summary(res$dist)
boxplot(res$dist)
plot(res$n, res$dist, pch = 20, main = "Kolmogorov distance vs. sample size",
     xlab = "sample size", ylab = "Kolmogorov distance",
     ylim = c(0, max(res$dist)))
uni.n <- min(res$n):max(res$n)
lines(uni.n, 1/(2*uni.n), col = "orange", lwd = 2)
legend("topright", legend = "minimal possible distance", fill = "orange")

## Illumina bead level data
library(beadarrayExampleData)
data(exampleBLData)
res <- KolmogorovMinDist(exampleBLData, Norm(), arrays = 1)
```

```
res1 <- KolmogorovMinDist(exampleBLData, Norm(), log = TRUE, arrays = 1)
summary(cbind(res$dist, res1$dist))
boxplot(list(res$dist, res1$dist), names = c("raw", "log-raw"))
sort(unique(res1$n))
plot(res1$n, res1$dist, pch = 20, main = "Kolmogorov distance vs. sample size",
     xlab = "sample size", ylab = "Kolmogorov distance",
     ylim = c(0, max(res1$dist)), xlim = c(min(res1$n), 56))
uni.n <- min(res1$n):56
lines(uni.n, 1/(2*uni.n), col = "orange", lwd = 2)
legend("topright", legend = "minimal possible distance", fill = "orange")
```

---

robloxbioc *Generic Function for Preprocessing Biological Data*

Description

Generic function for preprocessing biological data using optimally robust (rmx) estimators; confer Rieder (1994), Kohl (2005), Rieder et al (2008).

Usage

```
robloxbioc(x, ...)

## S4 method for signature 'matrix'
robloxbioc(x, eps = NULL, eps.lower = 0, eps.upper = 0.05, steps = 3L,
           fsCor = TRUE, mad0 = 1e-4)

## S4 method for signature 'AffyBatch'
robloxbioc(x, bg.correct = TRUE, pmcorrect = TRUE, normalize = FALSE,
           add.constant = 32, verbose = TRUE, eps = NULL,
           eps.lower = 0, eps.upper = 0.05, steps = 3L, fsCor = TRUE,
           mad0 = 1e-4, contrast.tau = 0.03, scale.tau = 10,
           delta = 2^(-20), sc = 500)

## S4 method for signature 'beadLevelData'
robloxbioc(x, channelList = list(greenChannel), probeIDs = NULL,
           useSampleFac = FALSE, sampleFac = NULL, weightNames = "wts",
           removeUnMappedProbes = TRUE, eps = NULL, eps.lower = 0,
           eps.upper = 0.05, steps = 3L, fsCor = TRUE, mad0 = 1e-4)
```

Arguments

x	biological data.
...	additional parameters.
eps	positive real ($0 < \text{eps} <= 0.5$): amount of gross errors. See details below.

eps.lower	positive real (0 <= eps.lower <= eps.upper): lower bound for the amount of gross errors. See details below.
eps.upper	positive real (eps.lower <= eps.upper <= 0.5): upper bound for the amount of gross errors. See details below.
steps	positive integer. k-step is used to compute the optimally robust estimator.
fsCor	logical: perform finite-sample correction. See function finiteSampleCorrection.
mad0	scale estimate used if computed MAD is equal to zero
bg.correct	if TRUE MAS 5.0 background correction is performed; confer bg.correct.mas.
pmcorrect	method used for PM correction; TRUE calls an algorithm which is comparable to the algorithm of MAS 5.0; confer pmcorrect.mas. If FALSE only the PM intensities are used.
normalize	logical: if TRUE, Affymetrix scale normalization is performed.
add.constant	constant added to the MAS 5.0 expression values before the normalization step. Improves the variance of the measure one no longer devides by numbers close to 0 when computing fold-changes.
verbose	logical: if TRUE, some messages are printed.
contrast.tau	a number denoting the contrast tau parameter; confer the MAS 5.0 PM correction algorithm.
scale.tau	a number denoting the scale tau parameter; confer the MAS 5.0 PM correction algorithm.
delta	a number denoting the delta parameter; confer the MAS 5.0 PM correction algorithm.
sc	value at which all arrays will be scaled to.
channelList	List of objects of class illuminaChannel that defines the summarisation to be performed where in our setup only the slots transFun and name have an effect on the computations. Setting the slots outlierFun, exprFun, and varFun has no effect. In any case rmx estimators are applied.
probeIDs	Vector of ArrayAddressIDs to be included in the summarized object. The default is to summarize all probes.
useSampleFac	if TRUE sections belonging to the same biological sample will be combined. The default is to summarize each section separately.
sampleFac	optional character vector giving which a sample identifer for each section
weightNames	name of column in the beadLevelData to take extract weights
removeUnMappedProbes	if TRUE and annotation information is stored in the beadLevelData object, any ArrayAddressIDs that cannot be mapped to ILMN IDs will be removed.

Details

The optimally-robust resp. the radius-minimax (rmx) estimator for normal location and scale is used to preprocess biological data. The computation uses a k-step construction with median and MAD as starting estimators; cf. Rieder (1994) and Kohl (2005).

If the amount of gross errors (contamination) is known, it can be specified by eps. The radius of the corresponding infinitesimal contamination neighborhood (infinitesimal version of Tukey's gross error model) is obtained by multiplying eps by the square root of the sample size.

If the amount of gross errors (contamination) is unknown, which is typically the case, try to find a rough estimate for the amount of gross errors, such that it lies between eps.lower and eps.upper.

If eps is NULL, the radius-minimax (rmx) estimator in sense of Rieder et al. (2001, 2008), respectively Section 2.2 of Kohl (2005) is used.

The algorithm used for Affymetrix data is similar to MAS 5.0 (cf. Affymetrix (2002)). The main difference is the substitution of the Tukey one-step estimator by our rmx k-step (k >= 1) estimator in the PM/MM correction step. The optional scale normalization is performed as given in Affymetrix (2002).

In case of Illumina data, the rmx estimator is used to summarize the bead types. The implementation for the most part copies summarize from **beadarray**.

For sample size <= 2, median and MAD are used for estimation.

If eps = 0, mean and sd are computed.

Value

Return value depends on the class of x. In case of "matrix" a matrix with columns "mean" and "sd" is returned. In case of "AffyBatch" an object of class "ExpressionSet" is returned. In case of "BeadLevelData" an object of class "ExpressionSetIllumina" is returned.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>,

update for beadarray versions >= 2.0.0 with support by Mark Dunnings and Andy Lynch

References

Affymetrix, Inc. (2002). *Statistical Algorithms Description Document*. Affymetrix, Santa Clara.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

Kohl M. and Deigner H.P. (2010). Preprocessing of gene expression data by optimally robust estimators. *BMC Bioinformatics*, 11:583.

M. Kohl, P. Ruckdeschel, and H. Rieder (2010). Infinitesimally Robust Estimation in General Smoothly Parametrized Models. *Statistical Methods and Application*, **19**(3):333-354.

Rieder, H. (1994) *Robust Asymptotic Statistics*. New York: Springer.

Rieder, H., Kohl, M. and Ruckdeschel, P. (2008) The Costs of not Knowing the Radius. *Statistical Methods and Applications* **17**(1) 13-40.

See Also

roblox, rowRoblox, AffyBatch-class, generateExprVal.method.mas, ExpressionSet-class, summarize

Examples

```
set.seed(123) # to have reproducible results for package checking

## similar to rowRoblox of package RobLox
ind <- rbinom(200, size=1, prob=0.05)
X <- matrix(rnorm(200, mean=ind*3, sd=(1-ind) + ind*9), nrow = 2)
robloxbioc(X)
robloxbioc(X, steps = 5)
robloxbioc(X, eps = 0.05)
robloxbioc(X, eps = 0.05, steps = 5)


## \donttest to reduce check time
## the function is designed for large scale problems
X <- matrix(rnorm(50000*20, mean = 1), nrow = 50000)
system.time(robloxbioc(X))

## using Affymetrix data
## confer example to generateExprVal.method.mas
## A more worked out example can be found in the scripts folder
## of the package.
data(SpikeIn)
probes <- pm(SpikeIn)
mas <- generateExprVal.method.mas(probes)
rl <- 2^robloxbioc(log2(t(probes)))
concentrations <- as.numeric(colnames(SpikeIn))
plot(concentrations, mas$exprs, log="xy", ylim=c(50,10000), type="b",
     ylab = "expression measures")
points(concentrations, rl[,1], pch = 20, col="orange", type="b")
legend("topleft", c("MAS", "roblox"), pch = c(1, 20))

## Affymetrix dilution data
library(affydata)
data(Dilution)
eset <- robloxbioc(Dilution)
## Affymetrix scale normalization
eset1 <- robloxbioc(Dilution, normalize = TRUE)

## Illumina bead level data
library(beadarrayExampleData)
data(exampleBLData)
res <- robloxbioc(exampleBLData, eps.upper = 0.5)
res
```

SimStudies *Perform Monte-Carlo Study.*

Description

The function `AffySimStudy` can be used to perform Monte-Carlo studies comparing Tukey's biweight and rmx estimators for normal location and scale. The function `IlluminaSimStudy` can be used to perform Monte-Carlo studies comparing Illumina's default method - a Huber-type skipped mean and sd (cf. Hampel (1985)) - and rmx estimators for normal location and scale. In addition, maximum likelihood (ML) estimators (mean and sd) and median and MAD are computed. The comparison is based on the empirical MSE.

Usage

```
AffySimStudy(n, M, eps, seed = 123, eps.lower = 0, eps.upper = 0.05,
             steps = 3L, fsCor = TRUE, contD, plot1 = FALSE,
             plot2 = FALSE, plot3 = FALSE)
IlluminaSimStudy(n, M, eps, seed = 123, eps.lower = 0, eps.upper = 0.05,
                 steps = 3L, fsCor = TRUE, contD, plot1 = FALSE,
                 plot2 = FALSE, plot3 = FALSE)
```

Arguments

`n`	integer; sample size, should be at least 3.
`M`	integer; Monte-Carlo replications.
`eps`	amount of contamination in [0, 0.5].
`seed`	random seed.
`eps.lower`	used by rmx estimator.
`eps.upper`	used by rmx estimator.
`steps`	integer; steps used for estimator construction.
`fsCor`	logical; use finite-sample correction.
`contD`	object of class `"UnivariateDistribution"`; contaminating distribution.
`plot1`	logical; plot cdf of ideal and real distribution.
`plot2`	logical; plot 20 (or M if M < 20) randomly selected samples.
`plot3`	logical; generate boxplots of the results.

Details

Normal location and scale with mean = 0 and sd = 1 is used as ideal model (without restriction due to equivariance).

Since there is no estimator which yields reliable results if 50 percent or more of the observations are contaminated, we use a modification where we re-simulate all samples including at least 50 percent contaminated data.

We use funtion `rowRoblox` for the computation of the rmx estimator.

Value

Data.frame including empirical MSE (standardized by sample size n) and relMSE with respect to the rmx estimator.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Affymetrix, Inc. (2002). *Statistical Algorithms Description Document*. Affymetrix, Santa Clara.

Hampel F.R. (1985). The breakdown points of the mean combined with some rejection rules. Technometrics, 27(2):95-107.

See Also

rowRoblox

Examples

```
set.seed(123) # to have reproducible results for package checking

AffySimStudy(n = 11, M = 100, eps = 0.02, contD = Norm(mean = 0, sd = 3),
             plot1 = TRUE, plot2 = TRUE, plot3 = TRUE)
IlluminaSimStudy(n = 30, M = 100, eps = 0.02, contD = Norm(mean = 0, sd = 3),
                 plot1 = TRUE, plot2 = TRUE, plot3 = TRUE)
```

Index