Package 'STARRS'

July 29, 2026

Type Package

Title Stochastic Robust Multivariate Statistics

Version 1.0

Date 2026-06-26

Description Algorithms for robust multivariate statistics (STochAstic Robust multivaRiate Statistics) including geometric median and geometric median covariance computation, k-medians clustering and robust median Principal Compenents Analysis (PCA), robust estimation of parameters for Gaussian, Student, or Laplace mixture models. 'STARRS' provides an independent, clean, consolidated and cohesive framework, while drawing inspiration from the approaches implemented in packages 'Gmedian', 'Kmedians', 'RGMM', 'RobRegression'.
Methods used in the package refer to
H. Robbins, S. Monro (1951) <doi:10.1214/aoms/1177729586>;
D. Kraus, V. M. Panaretos (2012) <doi:10.1093/biomet/ass037>;
H. Cardot, A. Godichon-Baggioni (2015) <doi:10.48550/arXiv.1504.02852>;
A. Godichon-Baggioni, S. Robin (2024) <doi:10.1007/s11222-023-10362-9>.

License GPL (>= 2)

Imports Rcpp (>= 1.0.13), foreach, mvtnorm, mclust, LaplacesDemon, genieclust, robustbase, RSpectra, ggplot2, reshape2, DescTools, checkmate, doFuture, future, parallel, capushe, bookdown, knitr

LinkingTo Rcpp, RcppArmadillo

LazyData true

VignetteBuilder knitr

Encoding UTF-8

Suggests rmarkdown

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Antoine Godichon-Baggioni [aut, cph], Stéphane Robin [aut], Daphné Giorgi [aut, cre]

Maintainer Daphné Giorgi <daphne.giorgi@sorbonne-universite.fr>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2026-07-29 17:00:19 UTC

Contents

`ASGMedian` *Averaged Stochastic Gradient geometric median computation*

Description

Compute the median of a matrix X using the Averaged Stochastic Gradient algorithm

$m_{k+1} = m_k + \gamma_{k+1} \frac{X_{k+1} - m_k}{\|X_{k+1} - m_k\|}$

$\bar{m}_{k+1} = \bar{m}_k + \frac{1}{k+1}(m_{k+1} - \bar{m}_k)$

Usage

```
ASGMedian(
  X,
  init_median = NULL,
  weights = NULL,
  gamma = 2,
  alpha = 0.75,
  nstart = 1L,
  epsilon = 1e-08
)
```

Arguments

X	A numerical matrix corresponding to the data of size $n \times p$. The rows are n observations in $\mathbf{R}^p$.
init_median	Initial value of the median, by default equal to 0 (internally handled)
weights	Vector of size n of the weights, by default equal to 1 (internally handled)
gamma	ASG parameter (2 by default)
alpha	ASG parameter (0.75 by default)
nstart	Number of starts for the ASG algorithm, by default 1
epsilon	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08

Value

The estimated median

Examples

```
N <- 1000
p <- 4
X <- matrix(rnorm(N*p),ncol=p)
ASGMedian(X)
```

ASGMedianCovariance *Averaged Stochastic Gradient geometric median covariation matrix computation*

Description

Compute the median covariation matrix of a matrix X using the Averaged Stochastic Gradient algorithm

$V_{k+1} = V_k + \gamma_{k+1} \frac{(X_{k+1}-\bar{m}_k)(X_{k+1}-\bar{m}_k)^T - V_n}{\|(X_{k+1}-\bar{m}_k)(X_{k+1}-\bar{m}_k)^T - V_n\|_F}$

$\bar{V}_{k+1} = \bar{V}_k + \frac{1}{k+1}(V_{k+1} - \bar{V}_k)$

Usage

```
ASGMedianCovariance(
  X,
  median_est = NULL,
  init_median_cov = NULL,
  weights = NULL,
  gamma = 2,
  alpha = 0.75,
  nstart = 1L
)
```

Arguments

X	A numerical matrix corresponding to the data of size $n \times p$. The rows are n observations in $\mathbf{R}^p$.
median_est	Estimation of the median m^*, typically an output of ASGMedian(X) (internally handled)
init_median_cov	Initial value of the median covariation matrix, by default equal to the identity matrix (internally handled)
weights	Vector of size n of the weights, by default equal to 1 (internally handled)
gamma	ASG parameter (2 by default)
alpha	ASG parameter (0.75 by default)
nstart	Number of starts for the ASG algorithm, by default 1

Value

The estimated median covariation matrix

Examples

```
N <- 1000
p <- 4
X <- matrix(rnorm(N*p),ncol=p)
med <- ASGMedian(X)
ASGMedianCovariance(X, median_est=med)
```

ASGRobustPCA *Robust PCA using ASG algorithm*

Description

Robust PCA using ASG algorithm for the median and the median covariation computation

Usage

```
ASGRobustPCA(
  X,
  init_median = rep(0, ncol(X)),
  init_median_cov = diag(ncol(X)),
  weights = rep(1, nrow(X)),
  epsilon = 1e-08,
  scores = 2,
  gamma = 2,
  alpha = 0.75,
  nstart = 1
)
```

Arguments

X	Matrix of size N*p corresponding to the data. The rows are the observations.
init_median	Initial value of the median, by default equal to 0
init_median_cov	Initial value of the median covariation matrix, by default equal to the identity matrix
weights	Vector of size N of the weights, by default equal to 1
epsilon	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
scores	Number of scores to compute, by default 2
gamma	ASG parameter (2 by default)
alpha	ASG parameter (0.75 by default)
nstart	Number of starts for the ASG algorithm, by default 1

Value

A list containing :

- median : the estimated median
- covmedian : the estimated median covariation matrix
- scores : the scores
- vectors: the eigen vectors

Examples

```
N <- 1000
p <- 4
X <- matrix(rnorm(N*p),ncol=p)
ASGRobustPCA(X)
```

`fixMC` *Fix point method for the estimation of the eigen values of the variance-covariance matrix*

Description

Given the eigen values δ of the median covariation matrix, this function estimates the eigen values λ of a variance-covariance matrix using the fix point method and Monte Carlo approximation.

Usage

```
fixMC(
  U,
  delta,
  init = NULL,
  niter = 10L,
  epsilon = 1e-08,
  out_index = as.integer(c())
)
```

Arguments

`U`	Matrix of size N*p corresponding to $\Sigma^{-1/2}(X - \mu)$. The rows are the observations. - In a gaussian model typically 'U = matrix(rnorm(N*p),ncol=p)' - In a Student model 'U <- matrix(rnorm(N*p)/sqrt(rchisq(1,df=df))*sqrt((df-2)), ncol=p))' - In a Laplace model 'U <- LaplacesDemon::rmvl(N,mu=rep(0,p),Sigma=diag(p))'
`delta`	Vector of size p of the eigen values of the median covariation matrix
`init`	Initial value of the vector of eigen values of the variance-covariance matrix, by default equal to delta (internally handled)
`niter`	Maximum number of iterations for the gradient descent algorithm, by default 10
`epsilon`	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
`out_index`	Indexes of the iterations for which we want to output the values of the estimates, default is the last iteration

Value

The vector of the estimated eigen values λ

Examples

```
delta <- c(1, 1)
N <- 1000
p <- length(delta)
U <- matrix(rnorm(N*p),ncol=p)
```

```
gradMC(U=U,delta=delta)
```

`gaussStudentCont_n5000_d5` *Gaussian data with Student contamination*

Description

We consider a Gaussian random vector $X \sim \mathcal{N}(0, \mathrm{diag}(1, \ldots, p))$ with $p = 5$, and generate a sample of size $n = 5000$. To investigate the robustness of the estimators, an increasing fraction of the observations, varying from 0% to 50%, is replaced by outliers drawn from a multivariate Student distribution and scaled by a factor of 10.

Usage

```
gaussStudentCont_n5000_d5
```

Format

List of contaminated (0%, 10%, 20%, 30%,40%,50%) data, each item being a list of 2 :

- `X` : $n \times p$ data matrix
- `C` : n-dimensional binary vector indicating which observation is an outlier

---

`GMMcontStudent` *Multivariate Gaussian data with Student contamination*

Description

Synthetic data set

A set of $n = 1500$ observations with dimension $d = 5$, equally spread into $K = 3$, each being sampled from a multivariate Gaussian distribution $\mathcal{N}_d(\mu_k, \Sigma_k)$ (with $k = 1, 2, 3$). The mean vectors are set to

$\mu_1 = 0_d, \mu_2 = 31_d, \mu_2 = -31_d$ (where 1_d stands for the d-dimensional vector filled with ones) and the variance matrices to $\Sigma_1 = 2\ I_d, \qquad \Sigma_2 = \mathrm{diag}([1, 2, \ldots d]), \qquad \Sigma_3 = \mathrm{diag}([1, 1/2, \ldots 1/d])$, ($I_d$ being the d-dimensional identity matrix).

In addition, we consider that a fraction $f = 30\%$ of the genuine observations have been replaced with observations arising from a mixture with K components of student distribution, with same respective location and scale parameters and with degree of freedom 1.

Usage

```
GMMcontStudent
```

Format

Output of the function `GMMcontStudent`

`gradMC` *Gradient descent for the estimation of the eigen values of the variance-covariance matrix*

Description

Given the eigen values δ of the median covariation matrix, this function estimates the eigen values λ of a variance-covariance matrix using the gradient descent method and Monte Carlo approximation.

Usage

```
gradMC(
  U,
  delta,
  init = NULL,
  niter = 10L,
  epsilon = 1e-08,
  step = as.numeric(c()),
  out_index = as.integer(c())
)
```

Arguments

`U`	Matrix of size N*p corresponding to $\Sigma^{-1/2}(X - \mu)$. The rows are the observations. - In a gaussian model typically 'U = matrix(rnorm(N*p),ncol=p)' - In a Student model 'U <- matrix(rnorm(N*p)/sqrt(rchisq(1,df=df))*sqrt((df-2)), ncol=p))' - In a Laplace model 'U <- LaplacesDemon::rmvl(N,mu=rep(0,p),Sigma=diag(p))'
`delta`	Vector of size p of the eigen values of the median covariation matrix
`init`	Initial value of the vector of eigen values of the variance-covariance matrix, by default equal to delta (internally handled)
`niter`	Maximum number of iterations for the gradient descent algorithm, by default 10
`epsilon`	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
`step`	Step of the gradient descent, by default 1
`out_index`	Indexes of the iterations for which we want to output the values of the estimates, default is the last iteration

Value

A list containing the estimated eigen values λ and the values of the estimates for different iterations

Examples

```
delta <- c(1, 1)
N <- 1000
p <- length(delta)
U <- matrix(rnorm(N*p),ncol=p)
gradMC(U=U,delta=delta)
```

`heteroGMMStudentCont_K3_d5_n1500`

Heteroscedastic setting gaussian GMM with Student contamination

---

Description

We consider a set of $n = 1500$ observations with dimension $d = 5$, equally spread into $K = 3$, each being sampled from a multivariate Gaussian distribution $\mathcal{N}_d(\mu_k, \Sigma_k)$ (with $k = 1, 2, 3$), with respective mean vector μ_k and variance matrix Σ_k. The mean vectors are set to $\mu_1 = \mathbf{0}_d, \mu_2 = 3\ \mathbf{1}_d, \mu_3 = -3\ \mathbf{1}_d$ (where $\mathbf{1}_d$ stands for the d-dimensional vector filled with ones) and the variance matrices to $\Sigma_1 = 2\ \mathbf{I}_d, \Sigma_2 = \text{diag}([1, 2, \dots d]), \Sigma_3 = \text{diag}([1, 1/2, \dots 1/d])$.

Usage

```
heteroGMMStudentCont_K3_d5_n1500
```

Format

List of contaminated (0%, 10%, 20%, 30%,40%,50%) data, each item being a list of 3 :

- X : $n \times p$ data matrix
- Z : n-dimensional vector indicating to which cluster each data belongs
- C : n-dimensional binary vector indicating which observation is an outlier

---

`homoGMMStudentCont_K3_d5_n1500`

Homoscedastic setting gaussian GMM with Student contamination

---

Description

We consider a set of $n = 1500$ observations with dimension $d = 5$, equally spread into $K = 3$, each being sampled from a multivariate Gaussian distribution $\mathcal{N}_d(\mu_k, \Sigma_k)$ (with $k = 1, 2, 3$), with respective mean vector μ_k and variance matrix Σ_k. The mean vectors are set to $\mu_1 = \mathbf{0}_d, \mu_2 = 3\ \mathbf{1}_d, \mu_3 = -3\ \mathbf{1}_d$ (where $\mathbf{1}_d$ stands for the d-dimensional vector filled with ones) and all variance matrices are equal: $\Sigma_1 = \Sigma_2 = \Sigma_3 = 2\mathbf{I}_d$.

Usage

```
homoGMMStudentCont_K3_d5_n1500
```

Format

List of contaminated (0%, 10%, 20%, 30%,40%,50%) data, each item being a list of 3 :

- X : $n \times p$ data matrix
- Z : n-dimensional vector indicating to which cluster each data belongs
- C : n-dimensional binary vector indicating which observation is an outlier

---

kmedians *K-medians algorithms*

Description

Perform k-medians clustering on a data matrix.

Usage

```
kmedians(
  X,
  nclust = 1:15,
  ninit = 0,
  niter = 20,
  method = "Offline",
  init = TRUE,
  nb_cores = 1
)
```

Arguments

`X`	A numerical matrix giving the data
`nclust`	A vector of positive integers giving the possible numbers of clusters. Default is `1:15`.
`ninit`	A non negative integer giving the number of random initializations. Default is `0`.
`niter`	A positive integer giving the number of iterations for the EM algorithms. Default is `20`.
`method`	The selected method for the K-medians algorithm. Can be • `Offline`: Default. Medians are computed with Weizsfeld algorithm. • `Semi-Online`: Medians are computed with ASG algorithm. • `Online`: Medians are computed with Online algorithm.
`init`	A logical argument telling if the function `'genie'` is used for initializing the algorithm. Default is `TRUE`.
`nb_cores`	Number of cores in parallel computation, by default 1

Value

List with the following components :

- `bestresults`: A list giving all the results for the clustering selected by `'capushe'`.
  - `cluster` A vector of positive integers giving the clustering.
  - `centers` A numerical matrix giving the centers of the clusters.
  - `SE` An integer giving the Sum of Errors.
- `allresults`: A list containing all the results.
- `SE`: A vector giving the Sum of Errors for each considered number of clusters.
- `cap`: The results given by the function `'capushe'` if `nclust` is of length larger than `10`.
- `Kopt`: An integer giving the number of clusters selected by `'capushe'` if `nclust` is of length larger than `10`.

References

Godichon-Baggioni, A. and Surendran, S. A penalized criterion for selecting the number of clusters for K-medians. *arxiv.org/abs/2209.03597*

Examples

```
n <- 1000
K <- 3
pcont <- 0.2
sphereGM <- rSphereGM(n=n,K=K,pcont=pcont)
X <-sphereGM$X
res <- kmedians(X)
plot(X, col=res$bestresult$cluster)
```

`multipleRobustMM` *Robust Mixture Model*

Description

Robust EM algorithm for Mixture Model for a number of clusters in a range

Usage

```
multipleRobustMM(
  X,
  nclust = 2:5,
  ninit = 10,
  init = "Mclust",
  methodMC = "RobbinsMC",
  methodMCM = "Weiszfeld",
```

```
  model = "Gaussian",
  df = NULL,
  niterWiesz = 50,
  niterEM = 50,
  niterMC = 50,
  sizeMC = 1000,
  logLikeInit = -1e+10,
  eps_dist = 1e-04,
  eps_eigval = 1e-10,
  eps_MCM_MC = 0.001,
  eps_tau = 1e-04,
  logPhiTol = -20,
  outlierThreshold = -20,
  gammaRMC = 0.75,
  cRMC = ncol(X),
  wRMC = 2,
  scale = FALSE,
  criterion = "BIC",
  nb_cores = parallel::detectCores()
)
```

Arguments

X	A numerical matrix giving the data of size nxd
nclust	Range of number of clusters
ninit	The number of random initializations (10 by default)
init	Kind of initialization, can be 'Mclust' (default) or 'genie'
methodMC	Method to compute the eigen values and vectors of variance-covariance matrix starting from median covariation matrix, can be "FixMC", "GradMC" or "RobbinsMC" (by default)
methodMCM	Method to compute the geometric median and the median covariation matrix, can be "Weiszfeld" (by default) or "ASG"
model	Type of distribution: can be "Gaussian" (by default), "Student" or "Laplace"
df	Degrees of freedom used when model is 'Student' (NULL by default). When used, it must be greater than 3
niterWiesz	Maximum number of iterations in the geometric median computation using the Weiszfeld algorithm
niterEM	Maximum number of updates in the EM algorithm
niterMC	Maximum number of updates in the gradient descent algorithm in the estimation of the eigen values of the variance-covariance matrix
sizeMC	Size of the sample for the Monte Carlo method in the estimation of the eigen values of the variance-covariance matrix
logLikeInit	Initial value of log likelihood
eps_dist	Distance between old and new centers after each iteration has to be greater than eps_dist

eps_eigval	Minimal value for eigen values (1e-10 by default)
eps_MCM_MC	Stopping criterion used in the Weiszfeld geometric median computation and in the estimation of the eigen values of the variance-covariance matrix
eps_tau	Small correction to avoid 0 and 1 values in tau computation (1e-04 by default)
logPhiTol	Tolerance for small values of log densities (-20 by default)
outlierThreshold	Threshold for outliers detection (-20 by default)
gammaRMC	Robbins-Monro parameter for eigen values estimation, if methodMC="RobbinsMC"
cRMC	Robbins-Monro parameter for eigen values estimation, if methodMC="RobbinsMC". When parameter scale=TRUE, cRMC=1 should be a good choice.
wRMC	Robbins-Monro parameter for eigen values estimation, if methodMC="RobbinsMC"
scale	Boolean value for scaling data before running the algorithm (FALSE by default)
criterion	Criterion for choice of best clustering, can be 'ICL' or 'BIC' (default)
nb_cores	Number of cores in parallel computation, by default parallel::detectCores()

Value

An object of class RMM, which is a list with the following components:

allresults A list containing the results from the RobustMM function for each number of classes specified in nclust.

bestresult A list corresponding to the output of RobustMM for the best model selected using the BIC or ICL criterion.

ICL A numeric vector of length length(nclust) containing the Integrated Complete-data Likelihood values for the different models.

BIC A numeric vector of length length(nclust) containing the Bayesian Information Criterion values for the different models.

data The original data matrix X.

nclust An integer vector specifying the number of classes (models) considered.

Kopt The optimal number of classes selected based on the BIC or ICL criterion.

References

https://cnrs.hal.science/hal-03853744/

Cardot, H., Cenac, P. and Zitt, P-A. (2013). Efficient and fast estimation of the geometric median in Hilbert spaces with an averaged stochastic gradient algorithm. *Bernoulli*, 19, 18-43.

Cardot, H. and Godichon-Baggioni, A. (2017). Fast Estimation of the Median Covariation Matrix with Application to Online Robust Principal Components Analysis. *Test*, 26(3), 461-480

Vardi, Y. and Zhang, C.-H. (2000). The multivariate L1-median and associated data depth. *Proc. Natl. Acad. Sci. USA*, 97(4):1423-1426.

See Also

robustMM

`offlineRobustVariance` *Offline robust estimation of the variance*

Description

Given a data set X, this function estimates robustly variance-covariance matrix via the geometric median and the Median Covariation Matrix.

Usage

```
offlineRobustVariance(
  X,
  methodMC = "RobbinsMC",
  methodMCM = "Weiszfeld",
  model = "Gaussian",
  df = NULL,
  computeOutliers = FALSE,
  cutoff = stats::qchisq(p = 0.95, df = ncol(X)),
  init_median = rep(0, ncol(X)),
  init_median_cov = diag(ncol(X)),
  niterWeisz = 50,
  niterMC = 50,
  sizeMC = 1000,
  epsMCM = 1e-08,
  epsMC = 1e-08,
  eps_eigval = 1e-04,
  eps_lambda = 1e-04,
  gammaMCM = 2,
  alphaMCM = 0.75,
  nstart = 1,
  gammaRMC = 0.75,
  cRMC = 2,
  wRMC = 2
)
```

Arguments

`X`	Matrix of size n*p corresponding to the data. The rows are the observations.
`methodMC`	Method to estimate the eigenvalues of the variance: can be "FixMC", "GradMC" and "RobbinsMC" (by default).
`methodMCM`	Method for estimating the Median Covariation Matrix: can be "Weiszfeld" (by default) or "ASG".
`model`	Type of distribution: can be "Gaussian" (by default), "Student" or "Laplace"
`df`	Number of degrees of freedom if model='Student'.
`computeOutliers`	Flag for outliers computation, FALSE by default.

cutoff	Threshold at which a data item is considered contaminated (if computeOutliers is TRUE). Default is $qchisq(p = 0.95, df = ncol(X))$.
init_median	A row vector for initializing the estimates of the median. Default is rep(0,ncol(X)).
init_median_cov	A matrix for initializing the estimates of the MCM. Default is Identity.
niterWeisz	The maximum number of iteration for the Weiszfeld algorithm. Default is 50.
niterMC	Maximum number of iterations is MethodMC='FixMC' or 'GradMC'.
sizeMC	Number of data generated in the Monte Carlo procedure to estimate the eigenvalues of the variance. Default is batch.
epsMCM	Stopping criterion for Weiszfeld algorithm: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
epsMC	Stopping criterion for MethodMC algorithm: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
eps_eigval	Minimal value for eigen values (1e-04 by default)
eps_lambda	Minimal value for estimated eigen values (1e-04 by default)
gammaMCM	A positive constant if methodMCM='ASG'.
alphaMCM	A positive constant if methodMCM='ASG'. Must belong to (0,1).
nstart	Number of run if methodMCM='ASG'.
gammaRMC	Robbins-Monro parameter for eigen values estimation, if methodMCM="RobbinsMC".
cRMC	Robbins-Monro parameter for eigen values estimation, if methodMCM="RobbinsMC".
wRMC	Robbins-Monro parameter for eigen values estimation, if methodMCM="RobbinsMC".

Value

A list with the following components:

variance A robust estimate of the variance.

outlier_labels If required, a binary vector of length `nrow(X)` indicating outliers (1 = outlier, 0 = inlier). Outlier detection is based on Mahalanobis distance.

distances If required, a numeric vector of length `nrow(X)` giving the Mahalanobis distances of each observation from the estimated center.

Examples

```
N <- 1000
p <- 10
X <- matrix(rnorm(N*p),ncol=p)
offlineRobustVariance(X=X)
```

`onlineRobustVariance` *Online robust estimation of the variance*

Description

Given a data set X, this function estimates online and robustly variance-covariance matrix via the geometric median and the Median Covariation Matrix.

Usage

```
onlineRobustVariance(
  X,
  nDataInit = 100,
  model = "Gaussian",
  df = NULL,
  computeOutliers = FALSE,
  cutoff = stats::qchisq(p = 0.95, df = ncol(X)),
  batch = ncol(X),
  niterWeisz = 100,
  sizeMC = batch,
  epsilon = 1e-08,
  eps_eigval = 1e-04,
  eps_lambda = 1e-04,
  gamma = 0.75,
  c = sqrt(ncol(X)),
  gammaRMC = 0.75,
  cRMC = ncol(X),
  wRMC = 2
)
```

Arguments

`X`	Matrix of size n*d corresponding to the data. The rows are the observations.
`nDataInit`	Number of data used for initializing the algorithm. Default is 100.
`model`	Type of distribution: can be "Gaussian" (by default), "Student" or "Laplace"
`df`	Number of degrees of freedom if model='Student'.
`computeOutliers`	Flag for outliers computation, FALSE by default. If TRUE, only Gaussian model is possible.
`cutoff`	Threshold at which a data item is considered contaminated (if computeOutliers is TRUE). Default is $qchisq(p = 0.95, df = ncol(X))$.
`batch`	Size of the batch. Default is ncol(X).
`niterWeisz`	Maximum number of iterations for first estimation of the median and the Median Covariation Matrix with the help of Weiszfeld algorithm. Default is 100.

sizeMC	Number of data generated in the Monte Carlo procedure to estimate the eigenvalues of the variance. Default is batch.
epsilon	Stopping criterion for RobbinsMC and Weisfeld algorithms: the algorithms stop when the difference between two iterations is less than epsilon, by default 1e-08
eps_eigval	Minimal value for eigen values (1e-04 by default)
eps_lambda	Minimal value for estimated eigen values (1e-04 by default)
gamma	Stochastic gradient parameter (0.75 by default). Must belong to (0,1).
c	Stochastic gradient parameter ($\sqrt{ncol(X)}$) by default). Must be positive.
gammaRMC	A parameter of the function RobbinsMC.
cRMC	A parameter of the function RobbinsMC.
wRMC	A parameter of the function RobbinsMC.

Value

A list with the following components:

variance A robust estimate of the variance.

outlier_labels If required, a binary vector of length `nrow(X)` indicating outliers (1 = outlier, 0 = inlier). Outlier detection is based on Mahalanobis distance.

distances If required, a numeric vector of length `nrow(X)` giving the Mahalanobis distances of each observation from the estimated center.

Examples

```
N <- 1000
p <- 10
X <- matrix(rnorm(N*p),ncol=p)
onlineRobustVariance(X=X)
```

`plot.RMM` *Plot method for Robust Mixture Model*

Description

Plot method for Robust Mixture Model

Usage

```
## S3 method for class 'RMM'
plot(
  x,
  showOutliers = FALSE,
 what = c("Two_Dim", "Two_Dim_Uncertainty", "ICL", "BIC", "Profiles", "Uncertainty"),
  K = NULL,
  ...
)
```

Arguments

x	An object of class RMM, typically an output of multipleRobustMM
showOutliers	Logical. If FALSE (by default), outliers are removed from the plot
what	Character vector. The type of plot to be displayed. Possible values are:
	"Two_Dim" Scatter plot of the projection of all the data points on the principal plan (along the two principal components) with color hue depending on clustering
	"Two_Dim_Uncertainty" Scatter plot of the projection of all the data points on the principal plan (along the two principal components) with color hue depending on clustering and size depending on Uncertainty
	"ICL" Plot of the ICL criterion as a function of the number of clusters
	"BIC" Plot of the BIC criterion as a function of the number of clusters
	"Profiles" Plot of the profiles of the data points with the median profile of each cluster
	"Uncertainty" Plot of the Uncertainty of the data points
K	Integer. The number of clusters to be displayed. If NULL (by default), the best result is displayed
...	Additional parameters

Value

A plot for each item in the what parameter

---

print.RMM *Printing of a robust mixture model output*

Description

print method for class RMM.

Usage

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

Arguments

x	Object of class RMM, typically an output of a call to multipleRobustMM function
...	Additional arguments affecting the summary produced.

Value

Print of output of a call to multipleRobustMM function

rGMMcontStudent *Generation of Gaussian mixture distribution with Student contamination*

Description

Function to generate a sample of a gaussian mixture model with Student contamination $X \sim \sum_{k=1}^{K} \pi_k Y_k$, where $Y_k \sim \mathcal{N}(\mu_k, \Sigma_k)$.

Usage

```
rGMMcontStudent(nk, muG, sigmaG, delta, dfT, muT, sigmaT)
```

Arguments

nk	Vector of the number of observations in each component, giving the proportions π_k
muG	Matrix of the means of the Gaussian components $\mu = (\mu_1, \ldots, \mu_K)$
sigmaG	Array of the variance-covariance matrices of the Gaussian components
delta	Proportion of contamination
dfT	Degrees of freedom of the Student contamination
muT	Matrix of the means of the Student contamination
sigmaT	Array of the variances of the Student contamination

Value

A list containing the sample X, the indicator of contamination C and the latent variable Z, giving the classification

Examples

```
# Dimensions
K <- 3; nk <- rep(500, K); d <- 5

# Gaussian parameters
mu <- rbind(mu1=rep(0, d), mu2=rep(3, d), mu3=rep(-3, d))
sigma <- array(dim=c(K, d, d))
sigma[1, , ] <- 2*diag(d); sigma[2, , ] <- diag(1:d); sigma[3, , ] <- diag(1/(1:d))

# Student contamination
fraction <- 0.3
dfT <- 1; dfT <- 1; muT <- mu; sigmaT <- sigma

# Simulation
GMMcontStudent <- rGMMcontStudent(nk=nk, muG=mu, sigmaG=sigma,
                                  delta=fraction,
                                  dfT=dfT, muT=muT, sigmaT=sigmaT)
```

`rGMMcontUniform` *Generation of Gaussian mixture distribution with uniform contamination*

Description

Function to generate a sample of a gaussian mixture model with uniform contamination $X \sim \sum_{k=1}^{K} \pi_k Y_k$, where $Y_k \sim \mathcal{N}(\mu_k, \Sigma_k)$.

Usage

```
rGMMcontUniform(nk, muG, sigmaG, delta, lUnif, uUnif)
```

Arguments

`nk`	Vector of the number of observations in each component, giving the proportions π_k
`muG`	Matrix of the means of the Gaussian components $\mu = (\mu_1, \ldots, \mu_K)$
`sigmaG`	Array of the variance-covariance matrices of the Gaussian components
`delta`	Proportion of contamination
`lUnif`	Vector of lower bounds of the uniform contamination
`uUnif`	Vector of upper bounds of the uniform contamination

Value

A list containing the sample X, the indicator of contamination C and the latent variable Z, giving the classification

Examples

```
# Dimensions
K <- 3; nk <- rep(500, K); d <- 5

# Gaussian parameters
mu <- rbind(mu1=rep(0, d), mu2=rep(3, d), mu3=rep(-3, d))
sigma <- array(dim=c(K, d, d))
sigma[1, , ] <- 2*diag(d); sigma[2, , ] <- diag(1:d); sigma[3, , ] <- diag(1/(1:d))

# Student contamination
fraction <- 0.3
lower <- rep(0,d); upper <- rep(1,d)

# Simulation
GMMcontUniform <- rGMMcontUniform(nk=nk, muG=mu, sigmaG=sigma,
                                  delta=fraction,
```

```
                                    lUnif=lower, uUnif=upper)
```

`robbinsMC` *Robbins-Monro method for the estimation of the eigen values of the variance-covariance matrix*

Description

Given the eigen values δ of the median covariation matrix, this function estimates the eigen values λ of a variance-covariance matrix using the Robbins-Monro method and Monte Carlo approximation.

Usage

```
robbinsMC(
  U,
  delta,
  init = NULL,
  gamma = 0.75,
  c = 2,
  w = 2,
  c_bar = 0,
  c_tilde = 0,
  sumlog = 1,
  epsilon = 1e-08
)
```

Arguments

`U`	Matrix of size N*p corresponding to $\Sigma^{-1/2}(X - \mu)$. The rows are the observations. - In a gaussian model typically 'U = matrix(rnorm(N*p),ncol=p)' - In a Student model 'U <- matrix(rnorm(N*p)/sqrt(rchisq(1,df=df))*sqrt((df-2)), ncol=p))' - In a Laplace model 'U <- LaplacesDemon::rmvl(N,mu=rep(0,p),Sigma=diag(p))'
`delta`	Vector of size p of the eigen values of the median covariation matrix
`init`	Initial value of the vector of eigen values of the variance-covariance matrix, by default equal to delta (internally handled)
`gamma`	Robbins-Monro parameter (0.75 by default)
`c`	Robbins-Monro parameter (2 by default)
`w`	Robbins-Monro parameter (2 by default)
`c_bar`	Hyperparameter for fine tuning. Do not change unless you are an experienced user (0 by default)
`c_tilde`	Hyperparameter for fine tuning. Do not change unless you are an experienced user (0 by default)

sumlog	Hyperparameter for fine tuning. Do not change unless you are an experienced user (1 by default)
epsilon	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08

Value

The vector of the estimated eigen values λ

Examples

```
delta <- c(1, 1)
N <- 1000
p <- length(delta)
U <- matrix(rnorm(N*p),ncol=p)
robbinsMC(U=U,delta=delta)
```

---

robustGMMOutput *Robust clustering output*

Description

Output of robust clustering on synthetic data set GMMcontStudent

Usage

```
robustGMMOutput
```

Format

Output of the call to the robust clustering function `multipleRobustMM(GMMcontStudent$X, nclust=1:5)`

---

robustMM *Robust Mixture Model*

Description

Robust EM algorithm for Mixture Model for a fixed number of clusters K

Usage

```
robustMM(
  X,
  K = 2,
  ninit = 10,
  initClust = NULL,
  methodMC = "RobbinsMC",
  methodMCM = "Weiszfeld",
  model = "Gaussian",
  df = NULL,
  niterWiesz = 50,
  niterEM = 50,
  niterMC = 50,
  sizeMC = 1000,
  logLikeInit = -1e+10,
  eps_dist = 1e-04,
  eps_eigval = 1e-10,
  eps_MCM_MC = 0.001,
  eps_tau = 1e-04,
  logPhiTol = -20,
  outlierThreshold = -20,
  gammaRMC = 0.75,
  cRMC = ncol(X),
  wRMC = 2,
  scale = FALSE,
  verbose = TRUE
)
```

Arguments

X	A numerical matrix giving the data of size nxd
K	Integer giving the number of clusters
ninit	The number of random initializations (10 by default)
initClust	Initial clustering, typically an output of `mclust::hclass` with a single number of clusters or `genieclust::genie` (NULL by default)
methodMC	Method to compute the eigen values and vectors of variance-covariance matrix starting from median covariation matrix, can be "FixMC", "GradMC" or "RobbinsMC" (by default)
methodMCM	Method to compute the geometric median and the median covariation matrix, can be "Weiszfeld" (by default) or "ASG"
model	Type of distribution: can be "Gaussian" (by default), "Student" or "Laplace"
df	Degrees of freedom used when model is 'Student' (NULL by default). When used, it must be greater than 3
niterWiesz	Maximum number of iterations in the geometric median computation using the Weiszfeld algorithm (50 by default)
niterEM	Maximum number of updates in the EM algorithm (50 by default)

niterMC	Maximum number of updates in the gradient descent algorithm in the estimation of the eigen values of the variance-covariance matrix (50 by default)
sizeMC	Size of the sample for the Monte Carlo method in the estimation of the eigen values of the variance-covariance matrix (1000 by default)
logLikeInit	Initial value of log likelihood (-1e10 by default).
eps_dist	Distance between old and new centers after each iteration has to be greater than eps_dist (1e-04 by default)
eps_eigval	Minimal value for eigen values (1e-10 by default)
eps_MCM_MC	Stopping criterion used in the Weiszfeld geometric median computation and in the estimation of the eigen values of the variance-covariance matrix
eps_tau	Small correction to avoid 0 and 1 values of tau (1e-04 by default)
logPhiTol	Tolerance for small values of log densities (-20 by default). This must be a negative value, since the log densities are first adjusted by substracting their max.
outlierThreshold	Threshold for outliers detection, on a log density criterion (-20 by default)
gammaRMC	Robbins-Monro parameter for eigen values estimation, if methodMC="RobbinsMC"
cRMC	Robbins-Monro parameter for eigen values estimation, if methodMC="RobbinsMC". When parameter scale=TRUE, cRMC=1 should be a good choice.
wRMC	Robbins-Monro parameter for eigen values estimation, if methodMC="RobbinsMC"
scale	Boolean value for scaling data before running the algorithm (FALSE by default)
verbose	Boolean value for printing execution messages

Value

A list with the following components:

centers A numeric matrix of size K × ncol(X), where each row corresponds to the center of a class.

Sigma A numeric array of size K × ncol(X) × ncol(X) containing the covariance matrices for each class.

LogLike The log-likelihood of the selected model.

entropy The entropy of the selected model.

tau A numeric matrix of size n × K, where each row contains the posterior probabilities of class membership for each observation.

clusters An integer vector of length nrow(X) indicating the assigned class for each observation.

niter The total number of iterations performed.

initEM The initialization used by the EM algorithm.

prop A numeric vector of length K giving the estimated proportion of each class.

outliers A binary vector of length nrow(X) where a value of 1 indicates that the corresponding observation is considered as an outlier. An observation is flagged as an outlier if its log-likelihood under all classes is below outlierThreshold.

References

https://cnrs.hal.science/hal-03853744/

Cardot, H., Cenac, P. and Zitt, P-A. (2013). Efficient and fast estimation of the geometric median in Hilbert spaces with an averaged stochastic gradient algorithm. *Bernoulli*, 19, 18-43.

Cardot, H. and Godichon-Baggioni, A. (2017). Fast Estimation of the Median Covariation Matrix with Application to Online Robust Principal Components Analysis. *Test*, 26(3), 461-480

Vardi, Y. and Zhang, C.-H. (2000). The multivariate L1-median and associated data depth. *Proc. Natl. Acad. Sci. USA*, 97(4):1423-1426.

See Also

multipleRobustMM

`rSphereGM` *Gaussian mixture on sphere sample generation*

Description

Generate a sample of a Gaussian Mixture Model whose centers are generate randomly on a sphere of a given radius.

Usage

```
rSphereGM(
  n = 500,
  d = 5,
  K = 3,
  pcont = 0,
  df = 1,
  cont = "Student",
  min = -5,
  max = 5,
  radius = 5
)
```

Arguments

`n`	Number of points per cluster
`d`	Dimension of the space
`K`	Number of clusters
`pcont`	Proportion of contamination
`df`	Degrees of freedom of the Student contamination
`cont`	Type of contamination: can be "Student" (default) or "Unif"
`min`	Minimum value of the contamination
`max`	Maximum value of the contamination
`radius`	Radius of the sphere on which the centers are generated

Value

A list containing the sample X and the clustering cluster

Examples

```
n <- 500
K <- 3
pcont <- 0.2

sphereGM <- rSphereGM(n=n,K=K,pcont=pcont)
```

`rTMMcontStudent` *Generation of Student mixture distribution with Student contamination*

Description

Function to generate a sample of a Student mixture model with Student contamination $X \sim \sum_{k=1}^{K} \pi_k Y_k$, where Y_k follow a Student distribution with means $\mu = (\mu_1, \ldots, \mu_K)$ and variance-covaraince matrices $\Sigma = (\Sigma_1, \ldots, \Sigma_K)$.

Usage

```
rTMMcontStudent(nk, dfT0, muT0, sigmaT0, delta, dfT1, muT1, sigmaT1)
```

Arguments

`nk`	Vector of the number of observations in each component, giving the proportions π_k
`dfT0`	Degrees of freedom of the Student distribution
`muT0`	Matrix of the means of the Student distribution
`sigmaT0`	Array of the variances of the Student distribution
`delta`	Proportion of contamination
`dfT1`	Degrees of freedom of the Student contamination
`muT1`	Matrix of the means of the Student contamination
`sigmaT1`	Array of the variances of the Student contamination

Value

A list containing the sample X, the indicator of contamination C and the latent variable Z, giving the classification

Examples

```
# Dimensions
K <- 3; nk <- rep(500, K); d <- 5

# Gaussian parameters
muT <- rbind(mu1=rep(0, d), mu2=rep(3, d), mu3=rep(-3, d))
sigmaT <- array(dim=c(K, d, d))
sigmaT[1, , ] <- 2*diag(d); sigmaT[2, , ] <- diag(1:d); sigmaT[3, , ] <- diag(1/(1:d))

# Student contamination
fraction <- 0.3
dfT <- 1;

# Simulation
TMMcontStudent <- rTMMcontStudent(nk=nk, dfT0=dfT, muT0=muT, sigmaT0=sigmaT,
                                  delta=fraction,
                                  dfT1=dfT, muT1=muT, sigmaT1=sigmaT)
```

`rTMMcontUniform` *Generation of Student mixture distribution with uniform contamination*

Description

Function to generate a sample of a Student mixture model with uniform contamination $X \sim \sum_{k=1}^{K} \pi_k Y_k$, where Y_k follow a Student distribution with means $\mu = (\mu_1, \ldots, \mu_K)$ and variance-covaraince matrices $\Sigma = (\Sigma_1, \ldots, \Sigma_K)$.

Usage

```
rTMMcontUniform(nk, dfT, muT, sigmaT, delta, lUnif, uUnif)
```

Arguments

`nk`	Vector of the number of observations in each component, giving the proportions π_k
`dfT`	Degrees of freedom of the Student distribution
`muT`	Matrix of the means of the Student distribution
`sigmaT`	Array of the variances of the Student distribution
`delta`	Proportion of contamination
`lUnif`	Vector of lower bounds of the uniform contamination
`uUnif`	Vector of uppers bounds of the uniform contamination

Value

A list containing the sample X, the indicator of contamination C and the latent variable Z, giving the classification

Examples

```
# Dimensions
K <- 3; nk <- rep(500, K); d <- 5

# Gaussian parameters
muT <- rbind(mu1=rep(0, d), mu2=rep(3, d), mu3=rep(-3, d))
sigmaT <- array(dim=c(K, d, d))
sigmaT[1, , ] <- 2*diag(d); sigmaT[2, , ] <- diag(1:d); sigmaT[3, , ] <- diag(1/(1:d))

# Student contamination
fraction <- 0.3
dfT <- 1;
lower <- rep(0,d); upper <- rep(1,d)

# Simulation
TMMcontUniform <- rTMMcontUniform(nk=nk, dfT=dfT, muT=muT, sigmaT=sigmaT,
                                  delta=fraction,
                                  lUnif=lower, uUnif=upper)
```

`summary.RMM` *Summary of robust mixture model output*

Description

Summary of the output of a Robust EM algorithm for Mixture Model for a number of clusters in a range

Usage

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

Arguments

`object`	Object of class `RMM`, typically an output of a call to multipleRobustMM function
`...`	Additional arguments affecting the summary produced

Value

Summary of output of a call to multipleRobustMM function

`WeiszfeldMedian` *Weiszfeld geometric median computation*

Description

Compute the median of a matrix X using the Weiszfeld algorithm

$m_{t+1} = \frac{\sum_{k=1}{}^{n} X_k / \|X_k - m_t\|}{\sum_{k=1}{}^{n} 1 / \|X_k - m_t\|}$

Usage

```
WeiszfeldMedian(
  X,
  init_median = NULL,
  weights = NULL,
  epsilon = 1e-08,
  nitermax = 100L
)
```

Arguments

`X`	A numerical matrix corresponding to the data of size $n \times p$. The rows are n observations in $\mathbf{R}^p$.
`init_median`	Initial value of the median, by default equal to 0 (internally handled)
`weights`	Vector of size n of the weights, by default equal to 1 (internally handled)
`epsilon`	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
`nitermax`	Maximum number of iterations for the Weiszfeld algorithm, by default 100

Value

The estimated median

Examples

```
N <- 1000
p <- 4
X <- matrix(rnorm(N*p),ncol=p)
WeiszfeldMedian(X)
```

`WeiszfeldMedianCovariance` *Weiszfeld geometric median covariation matrix computation*

Description

Compute the median covariation matrix of a matrix X using the Weiszfeld algorithm

$V_{t+1} = \frac{\sum_{k=1}{}^{n} \|(X_k - m^*)(X_k - m^*)^T - V_t\|_F^{-1}(X_k - m^*)(X_k - m^*)^T}{\sum_{k=1}{}^{n} \|(X_k - m^*)(X_k - m^*)^T - V_t\|_F^{-1}}$

Usage

```
WeiszfeldMedianCovariance(
  X,
  median_est = NULL,
  init_median_cov = NULL,
  weights = NULL,
  epsilon = 1e-08,
  nitermax = 100L
)
```

Arguments

`X`	A numerical matrix corresponding to the data of size $n \times p$. The rows are n observations in $\mathbf{R}^p$.
`median_est`	Estimation of the median m^*, by default `WeiszfeldMedian(X)` (internally handled)
`init_median_cov`	Initial value of the median covariation matrix, by default equal to the identity matrix (internally handled)
`weights`	Vector of size n of the weights, by default equal to 1 (internally handled)
`epsilon`	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
`nitermax`	Maximum number of iterations for the Weiszfeld algorithm, by default 100

Value

The estimated median covariation matrix

Examples

```
N <- 1000
p <- 4
X <- matrix(rnorm(N*p),ncol=p)
med_est <- WeiszfeldMedian(X)
WeiszfeldMedianCovariance(X,median_est=med_est)
```

`WeiszfeldRobustPCA` *Robust PCA using Weiszfeld algorithm*

Description

Robust PCA using Weiszfeld algorithm for the median and the median covariation computation

Usage

```
WeiszfeldRobustPCA(
  X,
  init_median = rep(0, ncol(X)),
  init_median_cov = (diag(ncol(X))),
  weights = rep(1, nrow(X)),
  scores = 2,
  epsilon = 1e-08,
  nitermax = 100
)
```

Arguments

`X`	Matrix of size N*p corresponding to the data. The rows are the observations.
`init_median`	Initial value of the median, by default equal to 0
`init_median_cov`	Initial value of the median covariation matrix, by default equal to the identity matrix
`weights`	Vector of size N of the weights, by default equal to 1
`scores`	Number of scores to compute, by default 2
`epsilon`	Stopping criterion: the algorithm stops when the difference between two iterations is less than epsilon, by default 1e-08
`nitermax`	Maximum number of iterations for the Weiszfeld algorithm, by default 100

Value

A list containing :

- `median` : the estimated median
- `covmedian` : the estimated median covariation matrix
- `scores` : the scores
- `vectors`: the eigen vectors

Examples

```
N <- 1000
p <- 4
X <- matrix(rnorm(N*p),ncol=p)
WeiszfeldRobustPCA(X)
```

Index