Description

The geometric median (also called spatial median or L1 median) is a robust multivariate indicator of central position. This library provides fast estimation procedures that can handle rapidly large samples of high dimensional data. Function \texttt{Gmedian} computes the geometric median of a numerical data set with averaged stochastic gradient algorithms, whereas \texttt{GmedianCov} computes the median covariation matrix, a useful indicator for robust PCA. Robust clustering, based on the geometric k-medians, can also be performed with the same type of recursive algorithm thanks to \texttt{kGmedian}. Less fast estimation procedures based on Weiszfeld’s algorithm are also available: function \texttt{Weiszfeld} computes the geometric median whereas \texttt{WeiszfeldCov} computes the median covariation matrix. These procedures may be preferred for small and moderate sample sizes. Note that weighting statistical units (for example with survey sampling weights) is allowed.

Details

Package: Gmedian
Type: Package
Title: Geometric Median, k-Median Clustering and Robust Median PCA
Version: 1.2.5
Date: 2020-03-02
Author: Herve Cardot
Maintainer: Herve Cardot <herve.cardot@u-bourgogne.fr>
Description: Fast algorithms for robust estimation with large samples of multivariate observations. Estimation of the geometric median, robust k-Gmedian clustering, and robust PCA based on the Gmedian covariation matrix.
License: GPL (>= 2)
Depends: R (>= 3.0.0)
Imports: Rcpp (>= 0.12.6), RSpectra, robustbase
LinkingTo: Rcpp, RcppArmadillo, RSpectra
NeedsCompilation: yes
Packaged: 2016-09-03 12:29:52 UTC; cardot
Repository: CRAN
Date/Publication: 2016-09-05 16:35:51

Index of help topics:

\begin{verbatim}
Gmedian             Gmedian
Gmedian-package     Geometric Median, k-Median Clustering and Robust Median PCA
GmedianCov          GmedianCov
Weiszfeld           Weiszfeld
WeiszfeldCov        WeiszfeldCov
kGmedian            kGmedian
\end{verbatim}
Gmedian

Author(s)

Herve Cardot

Maintainer: Herve Cardot <herve.cardot@u-bourgogne.fr>

References


Gmedian

Description

Computes recursively the Geometric median (also named spatial median or L1-median) with a fast averaged stochastic gradient algorithms that can deal rapidly with large samples of high dimensional data.

Usage

Gmedian(X, init = NULL, gamma = 2, alpha = 0.75, nstart=2, epsilon=1e-08)

Arguments

X           Data matrix, with n (rows) observations in dimension d (columns).
init        When NULL the starting point of the algorithm is the first observation. Else the starting point of the algorithm is provided by init.
gamma       Value (positive) of the constant controlling the descent steps (see details).
alpha       Rate of decrease of the descent steps (see details). Should satisfy $1/2 < alpha <= 1$.
nstart      Number of times the algorithm is ran over all the data set.
epsilon     Numerical tolerance. By default set to 1e-08.

Details

The recursive averaged algorithm is described in Cardot, Cenac, Zitt (2013), with descent steps defined as $\alpha_n = gamma / n^{alpha}$.
Value

Vector of the geometric median.

References


See Also

See also `GmedianCov`, `kGmedian` and `Weiszfeld`.

Examples

```r
## Simulated data - Brownian paths
n <- 1e4
d <- 500
x <- matrix(rnorm(n*d, sd=1/sqrt(d)), n, d)
x <- t(apply(x, 1, cumsum))

## Computation speed
system.time(replicate(10, {
  median.est = Gmedian(x)
}))

system.time(replicate(10, {
  mean.est = apply(x, 2, mean)
}))

## Accuracy with contaminated data
n <- 1e3
d <- 100
n.contaminated <- 50 ## 5% of contaminated observations
n.experiment <- 100
err.L2 <- matrix(NA, ncol=3, nrow=n.experiment)
colnames(err.L2) = c("mean (no contam.)", "mean (contam.)", "Gmedian")

for (n.sim in 1:n.experiment){
  x <- matrix(rnorm(n*d, sd=1/sqrt(d)), n, d)
x <- t(apply(x, 1, cumsum))
  err.L2[n.sim,1] <- sum((apply(x, 2, mean))^2/d)
  ind.contaminated <- sample(1:n, n.contaminated) ## contam. units
  x[ind.contaminated,] <- 5
  err.L2[n.sim,2] <- sum((apply(x, 2, mean))^2/d)
  err.L2[n.sim,3] <- sum(Gmedian(x)^2/d)
}

boxplot(err.L2, main="L2 error")
```
Description
Computes recursively the Geometric median and the (geometric) median covariation matrix with fast averaged stochastic gradient algorithms. The estimation of the Geometric median is performed first and then the median covariation matrix is estimated, as well as its leading eigenvectors. The original recursive estimator of the median covariation matrix may not be a non negative matrix. A fast projected estimator onto the convex closed cone of the non negative matrices allows to get a non negative solution.

Usage
GmedianCov(X, init=NULL, nn=TRUE, scores=2, gamma=2, gc=2, alpha=0.75, nstart=1)

Arguments
X
Data matrix, with n observations (rows) in dimension d (columns).

init
When NULL the starting point of the algorithm estimating the median is the first observation.

nn
When TRUE the algorithm provides a non negative estimates of the median covariation matrix. When nn=FALSE, the original algorithm is performed, with no guaranty that all the eigenvalues of the estimates are non negative

scores
An integer q, by default q=2. The function computes the eigenvectors of the median covariation matrix associated to the q largest eigenvalues and the corresponding principal component scores. No output if scores=0.

gamma
Value (positive) of the constant controling the descent steps (see details) for the algorithm computing median.

gc
Value (positive) of the constant controling the descent steps (see details) for algorithm computing the median covariation matrix

alpha
Rate of decrease of the descent steps, $1/2 < alpha <= 1$.

nstart
Number of time the algorithms are ran.

Details
The (fast) computation of the eigenvectors is performed by eigs_sym of package RSpectra. See Cardot, H. and Godichon-Baggioni (2017) for more details on the recursive algorithm. See also Gmedian. When nn=TRUE, the descent step is bounded above so that the solution remains non negative at each iteration. The principal components standard deviation is estimed robustly thanks to function scaleTau2 from package robustbase.
Value

- **median**: Vector of the geometric median
- **covmedian**: Median covariation matrix
- **vectors**: The \( q \) eigenvectors of the median covariation matrix associated to the \( q \) largest eigenvalues
- **scores**: Principal component scores corresponding to the \( q \) eigenvectors
- **sdev**: The \( q \) estimates of the standard deviation of the \( q \) principal components

References


See Also

See also `Gmedian` and `WeiszfeldCov`.

Examples

```r
## Simulated data - Brownian paths
n <- 1e3
d <- 50
x <- matrix(rnorm(n*d,sd=1/sqrt(d)), n, d)
x <- t(apply(x,1,cumsum))

## Estimation
median.est <- GmedianCov(x)
par(mfrow=c(1,2))
image(median.est$covmedian)  ## median covariation function
plot(c(1:d)/d,median.est$vectors[,1]*sqrt(d),type="l",xlab="Time",
     ylab="Eigenvectors",ylim=c(-1.4,1.4))
lines(c(1:d)/d,median.est$vectors[,2]*sqrt(d),lty=2)
```

Description

Fast k-medians clustering based on recursive averaged stochastic gradient algorithms. The procedure is similar to the `kmeans` clustering technique performed recursively with the MacQueen algorithm. The advantage of the kMedian algorithm compared to MacQueen strategy is that it deals with sum of norms instead of sum of squared norms, ensuring a more robust behaviour against outlying values.
**Usage**

```r
kGmedian(X, ncenters=2, gamma=1, alpha=0.75, nstart = 10, nstartkmeans = 10, iter.max = 20)
```

**Arguments**

- `X` matrix, with n observations (rows) in dimension d (columns).
- `ncenters` Either the number of clusters, say k, or a set of initial (distinct) cluster centres. If a number, the initial centres are chosen as the output of the `kmeans` function computed with the MacQueen algorithm.
- `gamma` Value of the constant controlling the descent steps (see details).
- `alpha` Rate of decrease of the descent steps.
- `nstart` Number of times the algorithm is ran, with random sets of initialization centers chosen among the observations.
- `nstartkmeans` Number of initialization points in the `kmeans` function for choosing the starting point of `kGmedian`.
- `iter.max` Maximum number of iterations considered in the `kmeans` function for choosing the starting point of `kGmedian`.

**Details**

See Cardot, Cenac and Monnez (2012).

**Value**

- `cluster` A vector of integers (from 1:k) indicating the cluster to which each point is allocated.
- `centers` A matrix of cluster centres.
- `withinrs` Vector of within-cluster sum of norms, one component per cluster.
- `size` The number of points in each cluster.

**References**


**See Also**

See also `Gmedian` and `kmeans`. 
Examples

# a 2-dimensional example
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

cl.kmeans <- kmeans(x, 2)
cl.kmedian <- kGmedian(x)

par(mfrow=c(1,2))
plot(x, col = cl.kmeans$cluster, main="kmeans")
points(cl.kmeans$centers, col = 1:2, pch = 8, cex = 2)

plot(x, col = cl.kmedian$cluster, main="kmedian")
points(cl.kmedian$centers, col = 1:2, pch = 8, cex = 2)

Description

Computes the Geometric median (also named spatial median or L1-median) with Weiszfeld’s algorithm.

Usage

Weiszfeld(X, weights = NULL, epsilon=1e-08, nitermax = 100)

Arguments

X    Data matrix, with n (rows) observations in dimension d (columns).
weights    When NULL, all observations have the same weight, say 1/n. Else, the user can provide a size n vector of weights (such as sampling weights). These weights are used in the estimating equation (see details).
epsilon    Numerical tolerance. By default 1e-08.
nitermax    Maximum number of iterations of the algorithm. By default set to 100.

Details

Weiszfeld’s algorithm (see Vardi and Zhang, 2000) is fast and accurate and can deal with large samples of high dimension data. However it is not as fast as the recursive approach proposed in Gmedian, which may be preferred for very large samples in high dimension. Weights can be given for statistical units, allowing to deal with data drawn from unequal probability sampling designs (see Lardin-Puech, Cardot and Goga, 2014).
Value

<table>
<thead>
<tr>
<th>median</th>
<th>Vector of the geometric median.</th>
</tr>
</thead>
<tbody>
<tr>
<td>iter</td>
<td>Number of iterations</td>
</tr>
</tbody>
</table>

References


See Also

See also `Gmedian` and `WeiszfeldCov`.

Examples

```r
## Robustness of the geometric median of n=3 points in dimension d=2.
a1 <- c(-1,0); a2 <- c(1,0); a3 <- c(0,1)
data.mat <- rbind(a1,a2,a3)
plot(data.mat,xlab="x",ylab="y")
med.est <- Weiszfeld(data.mat)
points(med.est$median,pch=19)

### weighted units
poids = c(3/2,1,1)
plot(data.mat,xlab="x",ylab="y")
med.est <- Weiszfeld(data.mat,weights=poids)
plot(data.mat,xlab="x",ylab="y")
points(med.est$median,pch=19)

### outlier
data.mat[3,] <- c(0,10)
plot(data.mat,xlab="x",ylab="y")
med.est <- Weiszfeld(data.mat)
points(med.est$median,pch=19)

### Computation speed
## Simulated data - Brownian paths
n <- 1e4
d <- 50
x <- matrix(rnorm(n*d,sd=1/sqrt(d)), n, d)
x <- t(apply(x,1,cumsum))

system.time(replicate(10, { median.est = Weiszfeld(x)})

system.time(replicate(10, { median.est = Gmedian(x)})

system.time(replicate(10, { mean.est = apply(x,2,mean)}))
```
Description

Estimation of the Geometric median covariation matrix with Weiszfeld’s algorithm. Weights (such as sampling weights) for statistical units are allowed.

Usage

`WeiszfeldCov(X, weights=NULL, scores=2, epsilon=1e-08, nitermax = 100)`

Arguments

- **X**: Data matrix, with n (rows) observations in dimension d (columns).
- **weights**: When NULL, all observations have the same weight, say 1/n. Else, the user can provide a size n vector of weights (such as sampling weights). These weights are used in the estimating equation (see details).
- **scores**: An integer q, by default q=2. The function computes the eigenvectors of the median covariation matrix associated to the q largest eigenvalues and the corresponding principal component scores. No output if scores=0.
- **epsilon**: Numerical tolerance. By default 1e-08.
- **nitermax**: Maximum number of iterations of the algorithm. By default set to 100.

Details

This fast and accurate iterative algorithm can deal with moderate size datasets. For large datasets use preferably `GmedianCov`, if fast estimations are required. Weights can be given for statistical units, allowing to deal with data drawn from unequal probability sampling designs (see Lardin-Puech, Cardot and Goga, 2014). The principal components standard deviation is estimated robustly thanks to function `scaleTau2` from package `robustbase`.

Value

- **median**: Vector of the geometric median
- **covmedian**: Median covariation matrix
- **vectors**: The scores=q eigenvectors of the median covariation matrix associated to the q largest eigenvalues
- **scores**: Principal component scores corresponding to the scores=q eigenvectors
- **sdev**: The scores=q robust estimates of the standard deviation of the principal components scores
- **iterm**: Number of iterations needed to estimate the median
- **itercov**: Number of iterations needed to estimate the median covariation matrix.
References


See Also

See also \texttt{Weiszfeld} and \texttt{GmedianCov}.

Examples

```r
## Simulated data - Brownian paths
n <- 1e3
d <- 50
x <- matrix(rnorm(n*d,sd=1/sqrt(d)), n, d)
x <- t(apply(x,1,cumsum))

## Estimation
median.est <- WeiszfeldCov(x)

par(mfrow=c(1,2))
image(median.est$covmedian) # median covariation function
plot(c(1:d)/d,median.est$vectors[,1]*sqrt(d),type="l",xlab="Time",
ylab="Eigenvectors",ylim=c(-1.4,1.4))
lines(c(1:d)/d,median.est$vectors[,2]*sqrt(d),lty=2)
```
Index

*Topic Gmedian
  Gmedian, 3
  Gmedian-package, 2
  GmedianCov, 5
  kGmedian, 6

*Topic WeiszfeldCov
  WeiszfeldCov, 10

*Topic Weiszfeld
  Weiszfeld, 8

eigs_sym, 5

Gmedian, 2, 3, 5–9
Gmedian-package, 2
GmedianCov, 2, 4, 5, 10, 11

kGmedian, 2, 4, 6
kmeans, 6, 7

scaleTau2, 5, 10

Weiszfeld, 2, 4, 8, 11
WeiszfeldCov, 2, 6, 9, 10