Package 'RiemGrassmann'

March 25, 2020

Type Package

Title Inference, Learning, and Optimization on Grassmann Manifold

Version 0.1.0

Description Grassmann manifold is a set of k-planes or linear subspaces in Euclidean space. We provide algorithms for statistical inference, optimization, and learning over the Grassmann manifold. For general exposition to the statistics on the manifold, see the book by Chikuse (2003) <doi:10.1007/978-0-387-21540-2>.

Encoding UTF-8

License GPL (>= 3)

Imports Rcpp, RiemBase, RiemBaseExt, utils, stats

LinkingTo Rcpp, RcppArmadillo, RiemBase

RoxygenNote 7.0.2

NeedsCompilation yes

Author Kisung You [aut, cre] (<https://orcid.org/0000-0002-8584-459X>)

Maintainer Kisung You <kyoustat@gmail.com>

Repository CRAN

Date/Publication 2020-03-25 15:50:06 UTC

R topics documented:

gr.hclust 2
gr.kmedoids 3
gr.mean 5
gr.pdist 6
gr.pdist2 8
package-RiemGrassmann 9

Index **10**

`gr.hclust` *Hierarchical Agglomerative Clustering on Grassmann Manifold*

Description

Given the `type` of distance measure and agglomeration scheme `method`, `gr.hclust` performs hierarchical clustering on Grassmann manifold using **fastcluster** package, which returns the same object as **stats** package's implementation while providing more efficient computation. See `hclust` for more details.

Usage

```
gr.hclust(
  input,
  type = c("Intrinsic", "Extrinsic", "Asimov", "Binet-Cauchy", "Chordal",
    "Fubini-Study", "Martin", "Procrustes", "Projection", "Spectral"),
  method = c("single", "complete", "average", "mcquitty", "ward.D", "ward.D2",
    "centroid", "median"),
  members = NULL
)
```

Arguments

`input`	either an array of size $(n \times k \times N)$ or a list of length N whose elements are $(n \times k)$ orthonormal basis (ONB) on Grassmann manifold.
`type`	type of distance measure. measure. Name of each type is *Case Insensitive* and *hyphen* can be omitted.
`method`	he agglomeration method to be used. This must be (an unambiguous abbreviation of) one of `"single"`, `"complete"`, `"average"`, `"mcquitty"`, `"ward.D"`, `"ward.D2"`, `"centroid"` or `"median"`.
`members`	`NULL` or a vector whose length equals the number of observations. See `hclust` for details.

Value

an object of class `hclust`. See `hclust` for details.

Author(s)

Kisung You

Examples

```
## generate a dataset with two types of Grassmann elements
#  group1 : first four columns of (8x8) identity matrix + noise
#  group2 : last  four columns of (8x8) identity matrix + noise
```

```
mydata = list()
sdval  = 0.25
diag8  = diag(8)
for (i in 1:10){
  mydata[[i]] = qr.Q(qr(diag8[,1:4] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}
for (i in 11:20){
  mydata[[i]] = qr.Q(qr(diag8[,5:8] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}

## try hierarchical clustering with "intrinsic" distance
opar <- par(no.readonly=TRUE)
hint <- gr.hclust(mydata, type="intrinsic")
plot(hint, main="intrinsic+single")
par(opar)


## do hierarchical clustering with different distance measures
alltypes = c("intrinsic","extrinsic","asimov","binet-cauchy",
"chordal","fubini-study","martin","procrustes","projection","spectral")
ntypes   = length(alltypes)

opar <- par(no.readonly=TRUE)
par(mfrow=c(2,5), pty="s")
for (i in 1:ntypes){
  hout = gr.hclust(mydata, type=alltypes[i])
  plot(hout, main=paste0("hclust::",alltypes[i]))
}
par(opar)
```

`gr.kmedoids` *k-Medoids Clustering on Grassmann Manifold*

Description

k-Medoids algorithm depends solely on the availability of concept that gives dissimilarity. We adopt `pam` algorithm from **cluster** package. See `pam` for more details.

Usage

```
gr.kmedoids(
  input,
  k = 2,
  type = c("Intrinsic", "Extrinsic", "Asimov", "Binet-Cauchy", "Chordal",
    "Fubini-Study", "Martin", "Procrustes", "Projection", "Spectral")
)
```

Arguments

input	either an array of size $(n \times k \times N)$ or a list of length N whose elements are $(n \times k)$ orthonormal basis (ONB) on Grassmann manifold.
k	the number of clusters
type	type of distance measure. measure. Name of each type is *Case Insensitive* and *hyphen* can be omitted.

Value

an object of class pam. See pam for details.

Author(s)

Kisung You

Examples

```
## generate a dataset with two types of Grassmann elements
#  group1 : first four columns of (8x8) identity matrix + noise
#  group2 : last  four columns of (8x8) identity matrix + noise

mydata = list()
sdval  = 0.25
diag8  = diag(8)
for (i in 1:10){
  mydata[[i]] = qr.Q(qr(diag8[,1:4] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}
for (i in 11:20){
  mydata[[i]] = qr.Q(qr(diag8[,5:8] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}

## do k-medoids clustering with 'intrinsic' distance
#  First, apply MDS for visualization
dmat = gr.pdist(mydata, type="intrinsic")
embd = stats::cmdscale(dmat, k=2)

#  Run 'gr.kmedoids' with different numbers of clusters
grint2 = gr.kmedoids(mydata, type="intrinsic", k=2)$clustering
grint3 = gr.kmedoids(mydata, type="intrinsic", k=3)$clustering
grint4 = gr.kmedoids(mydata, type="intrinsic", k=4)$clustering

#  Let's visualize
opar <- par(no.readonly=TRUE)
par(mfrow=c(1,3), pty="s")
plot(embd, pch=19, col=grint2, main="k=2")
plot(embd, pch=19, col=grint3, main="k=3")
plot(embd, pch=19, col=grint4, main="k=4")
par(opar)


## perform k-medoids clustering with different distance measures
```

```
#  iterate over all distance measures
alltypes = c("intrinsic","extrinsic","asimov","binet-cauchy",
"chordal","fubini-study","martin","procrustes","projection","spectral")
ntypes   = length(alltypes)
labels   = list()
for (i in 1:ntypes){
  labels[[i]] = gr.kmedoids(mydata, k=2, type=alltypes[i])$clustering
}

## visualize
#  1. find MDS scaling for each distance measure as well
embeds = list()
for (i in 1:ntypes){
  pdmat       = gr.pdist(mydata, type=alltypes[i])
  embeds[[i]] = stats::cmdscale(pdmat, k=2)
}

#  2. plot the clustering results
opar <- par(no.readonly=TRUE)
par(mfrow=c(2,5), pty="s")
for (i in 1:ntypes){
  pm = paste0("k-medoids::",alltypes[i])
  plot(embeds[[i]], col=labels[[i]], main=pm, pch=19)
}
par(opar)
```

`gr.mean` *Fréchet Mean on Grassmann Manifold*

Description

For manifold-valued data, Fréchet mean is the solution of following cost function,

$$\min_x \sum_{i=1}^{n} \rho^2(x, x_i), \quad x \in \mathcal{M}$$

for a given data $\{x_i\}_{i=1}^n$ and $\rho(x, y)$ is the geodesic distance between two points on manifold $\mathcal{M}$. It uses a gradient descent method with a backtracking search rule for updating.

Usage

```
gr.mean(x, type = c("intrinsic", "extrinsic"), eps = 1e-06, parallel = FALSE)
```

Arguments

`x`	either an array of size $(n \times k \times N)$ or a list of length N whose elements are $(n \times k)$ orthonormal basis (ONB) on Grassmann manifold.

`type`	type of geometry, either `"intrinsic"` or `"extrinsic"`.
`eps`	stopping criterion for the norm of gradient.
`parallel`	a flag for enabling parallel computation with OpenMP.

Value

a named list containing

mu an estimated mean matrix for ONB of size $(n \times k)$.

variation Fréchet variation with the estimated mean.

Author(s)

Kisung You

Examples

```
## generate a dataset with two types of Grassmann elements
#  first four columns of (8x8) identity matrix + noise
mydata = list()
sdval  = 0.1
diag8  = diag(8)
for (i in 1:10){
  mydata[[i]] = qr.Q(qr(diag8[,1:4] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}

## compute two types of means
mean.int = gr.mean(mydata, type="intrinsic")
mean.ext = gr.mean(mydata, type="extrinsic")

## visualize
opar <- par(no.readonly=TRUE)
par(mfrow=c(1,2))
image(mean.int$mu, main="intrinsic mean")
image(mean.ext$mu, main="extrinsic mean")
par(opar)
```

`gr.pdist` *Pairwise Distance for Data on Grassmann Manifold*

Description

For data on grassmann manifold $x_1, x_2, \ldots, x_N \in Gr(k,n)$, compute pairwise distances $d(x_i, x_j)$ via several metrics. The distance type `"intrinsic"` corresponds to geodesic distance while `"extrinsic"` is equivalent to `"chordal"` distance.

Usage

```
gr.pdist(
  input,
  type = c("Intrinsic", "Extrinsic", "Asimov", "Binet-Cauchy", "Chordal",
    "Fubini-Study", "Martin", "Procrustes", "Projection", "Spectral"),
  as.dist = TRUE,
  useR = FALSE
)
```

Arguments

input	either an array of size $(n \times k \times N)$ or a list of length N whose elements are $(n \times k)$ orthonormal basis (ONB) on Grassmann manifold.
type	type of distance measure. Name of each type is *Case Insensitive* and *hyphen* can be omitted.
as.dist	a logical; TRUE to return a dist object or FALSE to return an $(N \times N)$ symmetric matrix.
useR	a logical; TRUE to use R computations while FALSE goes everything in C++.

Value

a dist object or $(N \times N)$ symmetric matrix depending on as.dist.

Author(s)

Kisung You

Examples

```
## generate a dataset with two types of Grassmann elements
#  group1 : first four columns of (8x8) identity matrix + noise
#  group2 : last  four columns of (8x8) identity matrix + noise

mydata = list()
sdval  = 0.25
diag8  = diag(8)
for (i in 1:10){
  mydata[[i]] = qr.Q(qr(diag8[,1:4] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}
for (i in 11:20){
  mydata[[i]] = qr.Q(qr(diag8[,5:8] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}

## try 'intrinsic' distance using C++ implementation
dmat = gr.pdist(mydata, type="intrinsic", as.dist=FALSE)
opar = par(no.readonly=TRUE)
par(pty="s")
image(dmat, main="intrinsic distance")
par(opar)
```

```
## compute and visualize distances for all types
#  we will iterate over all measures
alltypes = c("intrinsic","extrinsic","asimov","binet-cauchy",
"chordal","fubini-study","martin","procrustes","projection","spectral")
ntypes   = length(alltypes)

opar <- par(no.readonly=TRUE)
par(mfrow=c(2,5), pty="s")
for (i in 1:ntypes){
  dmat = gr.pdist(mydata, type=alltypes[i], as.dist=FALSE)
  image(dmat[,20:1], main=alltypes[i])
}
par(opar)
```

`gr.pdist2` *Pairwise Distance for Two Sets Data on Grassmann Manifold*

Description

For data on grassmann manifold $x_1, x_2, \ldots, x_M \in Gr(k,n)$ and $y_1, y_2, \ldots, y_N \in Gr(k,n)$, compute pairwise distances $d(x_i, y_j)$ via several metrics. The distance type `"intrinsic"` corresponds to geodesic distance while `"extrinsic"` is equivalent to `"chordal"` distance.

Usage

```
gr.pdist2(
  input1,
  input2,
  type = c("Intrinsic", "Extrinsic", "Asimov", "Binet-Cauchy", "Chordal",
    "Fubini-Study", "Martin", "Procrustes", "Projection", "Spectral"),
  useR = FALSE
)
```

Arguments

`input1`	either an array of size $(n \times k \times M)$ or a list of length M whose elements are $(n \times k)$ orthonormal basis (ONB) on Grassmann manifold.
`input2`	either an array of size $(n \times k \times N)$ or a list of length N whose elements are $(n \times k)$ orthonormal basis (ONB) on Grassmann manifold.
`type`	type of distance measure. Name of each type is *Case Insensitive* and *hyphen* can be omitted.
`useR`	a logical; `TRUE` to use R computations while `FALSE` goes everything in C++.

Value

an $(M \times N)$ matrix of pairwise distances.

Author(s)

Kisung You

Examples

```
## generate a dataset with two types of Grassmann elements
#  group1 : first four columns of (8x8) identity matrix + noise
#  group2 : last  four columns of (8x8) identity matrix + noise

mydata1 = list()
mydata2 = list()
sdval   = 0.25
diag8   = diag(8)
for (i in 1:10){
  mydata1[[i]] = qr.Q(qr(diag8[,1:4] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}
for (i in 1:10){
  mydata2[[i]] = qr.Q(qr(diag8[,5:8] + matrix(rnorm(8*4,sd=sdval),ncol=4)))
}



## compute and visualize distances for all types
#  we will iterate over all measures
alltypes = c("intrinsic","extrinsic","asimov","binetcauchy",
"chordal","fubinistudy","martin","procrustes","projection","spectral")
ntypes   = length(alltypes)

opar <- par(no.readonly=TRUE)
par(mfrow=c(2,5), pty="s")
for (i in 1:ntypes){
  dmat = gr.pdist2(mydata1, mydata2, type=alltypes[i])
  image(dmat, main=alltypes[i])
}
par(opar)
```

`package-RiemGrassmann` *Inference, Learning, and Optimization on Grassmann Manifold*

Description

Grassmann manifold $Gr(k,n)$ is the set of k-planes, or k-dimensional subspaces in R^n, which is indeed a compact Riemannian manifold. In this package, we use a convention that each element in $Gr(k,n)$ is represented as an orthonormal basis (ONB) $X \in \mathbf{R}^{n \times k}$ where

$$X^\top X = I_k$$

.

Index

dist, *7*

gr.hclust, 2
gr.kmedoids, 3
gr.mean, 5
gr.pdist, 6
gr.pdist2, 8

hclust, *2*

package-RiemGrassmann, 9
pam, *3*, *4*