Package 'pivmet'

August 27, 2018

Type Package

Title Pivotal methods for relabelling and clustering

Version 0.1.0

Author Leonardo Egidi

Maintainer The package maintainer <legidi@units.it>

Description More about what it does (maybe more than one line)
Use four spaces when indenting paragraphs within the Description.

License What license is it under?

Encoding UTF-8

LazyData true

LazyLoad yes

Depends R, mvtnorm, bayesmix, RcmdrMisc,
cluster, mclust, runjags, rjags, MASS

Suggests knitr

VignetteBuilder knitr

RemoteType github

RemoteHost https://api.github.com

RemoteRepo pivmet

RemoteUsername LeoEgidi

RemoteRef master

RemoteSha 59e8616331302cc911ae13b6190580dcfee099b9

GithubRepo pivmet

GithubUsername LeoEgidi

GithubRef master

GithubSHA1 59e8616331302cc911ae13b6190580dcfee099b9

RoxygenNote 6.0.1

BuildManual yes

R topics documented:

`MUS` *MUS algorithm*

Description

Finding the pivotal units through a sequential search in the symmetric matrix C

Usage

```
MUS(C, clusters, prec_par)
```

Arguments

`C`	Square symmetrix matrix with value bounded in `[0,1]`. For instance, a co-association matrix resulting from clustering ensembles.
`clusters`	An initial group assignment for the `N` statistical units in `k` groups.
`prec_par`	A precision parameter for exploring a greater number of algorithm solutions. Default value is 5.

Details

See the vignette.

Value

`maxima`	The `k` maxima units

Examples

```
N <- 620
centers  <- 3
n1 <- 20
n2 <- 100
n3 <- 500
# generate data
x  <- matrix(NA, N,2)
truegroup <- c( rep(1,n1), rep(2, n2), rep(3, n3))
for (i in 1:n1){
 x[i,]=rmvnorm(1, c(1,5), sigma=diag(2))}
for (i in 1:n2){
 x[n1+i,]=rmvnorm(1, c(4,0), sigma=diag(2))}
```

```
for (i in 1:n3){
 x[n1+n2+i,]=rmvnorm(1, c(6,6), sigma=diag(2))}
H <- 1000
a <- matrix(NA, H, N)

for (h in 1:H){
   a[h,] <- kmeans(x,centers)$cluster
}
# build the similarity matrix
sim_matr <- matrix(1, N,N)
for (i in 1:(N-1)){
  for (j in (i+1):N){
     sim_matr[i,j] <- sum(a[,i]==a[,j])/H
     sim_matr[j,i] <- sim_matr[i,j]
     }
}

cl <- KMeans(x, centers)$cluster
mus_alg <- MUS(C = sim_matr, clusters = cl, prec_par = 5)
```

`piv_KMeans` *K-means Clustering Using MUS algorithm*

Description

Perform k-means clustering on a data matrix using MUS algorithm for seeding initialization.

Usage

```
piv_KMeans(x, centers, piv.criterion, iter.mus, prec.par, alg.type, iter.max,
  num.seeds)
```

Arguments

`x`	A numeric matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a dataframe with all numeric columns).
`centers`	The number of clusters in the solution.
`piv.criterion`	The pivotal criterion used for detecting pivotal units. If `centers <= 4`, default method is `MUS`. If `centers > 4`, the user may choose among the following: `maxsumint, maxsumnoint, maxsumdiff`.
`iter.mus`	The number of different ensembles for the MUS algorithm (if NULL, default is 1000)
`prec.par`	The precision parameter used in the MUS algorithm
`alg.type`	The type of clustering used for the initial seeding. Possible choices: `kmeans`, `KMeans`, `hclust`.
`iter.max`	The maximum number of iterations allowed.
`num.seeds`	The number of different starting random seeds to use. Each random seed results in a different k-means solution.

Value

A list with components

`cluster`	A vector of integers indicating the cluster to which each point is allocated.
`centers`	A matrix of cluster centres (centroids).
`totss`	The total sum of squares.
`withinss`	The within-cluster sum of squares for each cluster.
`tot.withinss`	The within-cluster sum of squares summed across clusters.
`betwennss`	The between-cluster sum of squared distances.
`size`	The number of points in each cluster.
`iter`	The number of (outer) iterations.
`ifault`	integer: indicator of a possible algorithm problem – for experts.
`pivots`	The pivotal units identified by the MUS algorithm

Author(s)

Leonardo Egidi legidi@units.it

Examples

```
n  <- 620
k  <- 3
n1 <- 20
n2 <- 100
n3 <- 500
x  <- matrix(NA, n,2)
truegroup <- c( rep(1,n1), rep(2, n2), rep(3, n3))

for (i in 1:n1){
 x[i,]=rmvnorm(1, c(1,5), sigma=diag(2))}
for (i in 1:n2){
 x[n1+i,]=rmvnorm(1, c(4,0), sigma=diag(2))}
for (i in 1:n3){
 x[n1+n2+i,]=rmvnorm(1, c(6,6), sigma=diag(2))}

res <- piv_KMeans(x, k)


par(mfrow=c(1,2), pty="s")
colors_cluster <- c("grey", "darkolivegreen3", "coral")
colors_centers <- c("black", "darkgreen", "firebrick")
plot(x, col = colors_cluster[truegroup],
   bg= colors_cluster[truegroup], pch=21, xlab="x[,1]",
   ylab="x[,2]", cex.lab=1.5,
   main="True data", cex.main=1.5)

plot(x, col = colors_cluster[res$cluster],
   bg=colors_cluster[res$cluster], pch=21, xlab="x[,1]",
   ylab="x[,2]", cex.lab=1.5,
   main="MUSK-means", cex.main=1.5)
points(x[res$pivots[1],1], x[res$pivots[1],2],
   pch=24, col=colors_centers[1],bg=colors_centers[1],
```

```
   cex=1.5)
points(x[res$pivots[2],1], x[res$pivots[2],2],
   pch=24,  col=colors_centers[2], bg=colors_centers[2],
   cex=1.5)
points(x[res$pivots[3],1], x[res$pivots[3],2],
   pch=24, col=colors_centers[3], bg=colors_centers[3],
   cex=1.5)
points(res$centers, col = colors_centers[1:k],
   pch = 8, cex = 2)
```

`piv_MCMC` *JAGS Sampling for Gaussian Mixture Models and Clustering via Co-Association Matrix.*

Description

Perform MCMC JAGS sampling for Gaussian mixture models, post-process the chains and apply a clustering technique to the MCMC sample. Pivotal units for each group are selected among four alternative criteria.

Usage

```
piv_MCMC(y, k, nMC, piv.criterion, clustering)
```

Arguments

`y`	N-dimensional data vector/matrix.
`k`	Number of mixture components.
`nMC`	Number of MCMC iterations for the JAGS function execution.
`piv.criterion`	The pivotal method used for detecting the pivots, one for each group. Possible choices: `maxsumint`, `maxsumnoint`, `maxsumdiff`, `MUS`. `MUS` is available for `k<5`. If `piv.criterion=NULL`, `maxsumdiff` is chosen by default. See the vignette for a thorough and detailed list of available pivotal methods.
`clustering`	The clustering technique adopted for partitioning the `N` observations into `k` groups. Possible choices: `diana` (default), `hclust`.

Details

The function fits a Bayesian Gaussian mixture model of the form:

$$(Y_i|Z_i = j) \sim f(y; \mu_j, \phi),$$

where the Z_i, $i = 1, \ldots, n$, are i.i.d. random variables, $j = 1, \ldots, k$, ϕ is a parameter which is common to all components, $Z_i \in 1, \ldots, k$, and

$$P(Z_i = k) = \pi_k.$$

The likelihood of the model is then

$$L(y; \mu, \pi, \phi) = \prod_{i=1}^{n} \sum_{j=1}^{k} \pi_k f(y_i; \mu_k, \phi),$$

with $\mu = (\mu_1, \ldots, \mu_k)$ component-specific parameters and $\pi = (\pi_1, \ldots, \pi_k)$ mixture weights. Let ν denote a permutation of $1, \ldots, k$, and let $\nu(\mu) = (\mu_{\nu(1)}, \ldots, \mu_{\nu(k)})$, $\nu(\pi) = (\pi_{\nu(1)}, \ldots, \pi_{\nu(k)})$ be the corresponding permutations of μ and π. Denote by V the set of all the permutations of the indexes $1, \ldots, k$, the likelihood above is invariant under any permutation $\nu \in V$, that is

$$L(y; \mu, \pi, \phi) = L(y; \nu(\mu), \nu(\pi), \phi).$$

As a consequence, the model is unidentified with respect to an arbitrary permutation of the labels. When Bayesian inference for the model is performed, if the prior distribution $p_0(\mu, \pi, \phi)$ is invariant under a permutation of the indices, then so is the posterior. That is, if $p_0(\mu, \pi, \phi) = p_0(\nu(\mu), \nu(\pi), \phi)$, then

$$p(\mu, \pi, \phi|y) \propto p_0(\mu, \pi, \phi)L(y; \mu, \pi, \phi)$$

is multimodal with (at least) $k!$ modes. The function performs JAGS sampling using the `bayesmix` package for univariate Gaussian mixtures, and the `runjags` package for bivariate Gaussian mixtures. After MCMC sampling, this function calls the `piv_sel()` function and yields the pivots obtained from one among four different methods: `maxsumint`, `maxsumnoint`, `maxsumdiff` and `MUS` (available only if `k < 5`) (see the vignette for thorough details)

Value

The function gives the MCMC output, the clustering solutions and the pivotal indexes. Here is a complete list of outputs.

`Freq`	Number of units corresponding to each group for the post-processed chains.
`z`	Post-processed latent vector.
`ris`	MCMC output array as provided by JAGS.
`groupPost`	Post-processed group vector.
`mu_switch`	Post-processed MCMC chains for the mean parameters.
`mu_pre_switch_compl`	Pre-precessed MCMC chains for the mean parameters.
`C`	Co-association matrix constructed from the MCMC sample.
`grr`	Group vector allocation as provided by `diana` or `hclust`.
`clust_sel`	clustering solution obtained via `diana` or `hclust` function.
`true.iter`	The number of MCMC iterations for which their number of groups exactly coincides with the prespecified number of groups `k`.

Author(s)

Leonardo Egidi legidi@units.it

References

Egidi, L., Pappada, R., Pauli, F. and Torelli, N. (2018). Relabelling in Bayesian Mixture Models by Pivotal Units. Statistics and Computing, 28(4), 957-969, DOI 10.1007/s11222-017- 9774-2.

Examples

```
N   <- 200
k   <- 4
nMC <- 1000
M1  <-c(-.5,8)
M2  <- c(25.5,.1)
M3  <- c(49.5,8)
M4  <- c(63.0,.1)
Mu  <- matrix(rbind(M1,M2,M3,M4),c(4,2))
stdev    <- cbind(rep(1,k), rep(200,k))
Sigma.p1 <- matrix(c(stdev[1,1],0,0,stdev[1,1]), nrow=2, ncol=2)
Sigma.p2 <- matrix(c(stdev[1,2],0,0,stdev[1,2]), nrow=2, ncol=2)
W <- c(0.2,0.8)
sim <- piv_sim(N,k,Mu, stdev, Sigma.p1,Sigma.p2,W)
res <- piv_MCMC(sim$y, k, nMC)


# Fishery data (bayesmix package)

data(fish)
y <- fish[,1]
k <- 5
nMC <- 5000
res <- piv_MCMC(y, k, nMC)
```

---

piv_plot *Plotting outputs from pivotal relabelling*

Description

Plot and visualize MCMC outputs, posterior relabelled chains and estimates and diagnostics.

Usage

```
piv_plot(y, mcmc, rel_est, type)
```

Arguments

y	Data vector or matrix.
mcmc	The ouptut of the raw MCMC sampling.
rel_est	Pivotal estimates as provided by piv_rel.
type	Type of plots required. Choose among: "chains", "estimates", "estimates_hist".

Examples

```
# Fishery data

data(fish)
y <- fish[,1]
N <- length(y)
k <- 5
```

```
nMC <- 5000
res <- piv_MCMC(y, k, nMC)
rel <- piv_rel(mcmc=res, nMC = nMC)
piv_plot(y, res, rel, "chains")
piv_plot(y, res, rel, "estimates")
piv_plot(y, res, rel, "estimates_hist")
```

`piv_rel` *Perfroming the pivotal relabelling step and computing the relabelled posterior estimates*

Description

This function allows to perform the pivotal relabelling procedure described in Egidi et al. (2018) and to obtain the relabelled posterior estimates.

Usage

```
piv_rel(mcmc, nMC)
```

Arguments

`mcmc`	The output of the MCMC sampling from `piv_MCMC`.
`nMC`	The number of total MCMC iterations (given in input to the `piv_MCMC` function, or any function suited for MCMC sampling).

Details

Prototypical models in which the label switching problem arises are mixture models, where for a sample $y = (y_1, \ldots, y_n)$ we assume

$$(Y_i|Z_i = j) \sim f(y; \mu_j, \phi),$$

where the Z_i, $i = 1, \ldots, n$, are i.i.d. random variables, $j = 1, \ldots, k$, ϕ is a parameter which is common to all components, $Z_i \in 1, \ldots, k$, and

$$P(Z_i = k) = \pi_k.$$

This model is unidentified with respect to an arbitrary permutation of the labels $1, ..., k$. Relabelling means permuting the labels at each iteration of the Markov chain in such a way that the relabelled chain can be used to draw inferences on component-specific parameters.

We assume here that an MCMC sample is obtained from the posterior distribution for model above– for instance via `piv_MCMC` function–with a prior distribution which is labelling invariant. Furthermore, suppose that we can find k units, one for each group, which are (pairwise) separated with (posterior) probability one (that is, the posterior probability of any two of them being in the same group is zero). It is then straightforward to use the k units, called pivots in what follows, to identify the groups and to relabel the chains (see the vignette for thorough details).

Value

This function gives the relabelled posterior estimates–both mean and medians–obtained from the Markov chains of the MCMC sampling.

`mu_rel_mean`	Estimated posterior means
`mu_rel_median`	Estimated posterior medians
`mu_rel_complete`	Complete relabelled chains
`Final_It`	The final number of valid iterations

Author(s)

Leonardo Egidi `legidi@units.it`

References

Egidi, L., Pappada, R., Pauli, F. and Torelli, N. (2018). Relabelling in Bayesian Mixture Models by Pivotal Units. Statistics and Computing, 28(4), 957-969, DOI 10.1007/s11222-017- 9774-2.

Examples

```
#Univariate simulation

N   <- 250
nMC <- 2500
k   <- 3
p   <- rep(1/k,k)
x   <- 3
stdev <- cbind(rep(1,k), rep(200,k))
Mu    <- seq(-trunc(k/2)*x,trunc(k/2)*x,length=k)
W     <- c(0.2,0.8)
sim   <- piv_sim(N,k,Mu,stdev,W=W)
res   <- piv_MCMC(sim$y, k, nMC)
rel   <- piv_rel(mcmc=res, nMC = nMC)


#Bivariate simulation

N <- 200
k <- 3
nMC <- 5000
M1  <- c(-.5,8)
M2  <- c(25.5,.1)
M3  <- c(49.5,8)
Mu  <- matrix(rbind(M1,M2,M3),c(k,2))
stdev <- cbind(rep(1,k), rep(200,k))
Sigma.p1 <- matrix(c(stdev[1,1],0,0,stdev[1,1]),
                   nrow=2, ncol=2)
Sigma.p2 <- matrix(c(stdev[1,2],0,0,stdev[1,2]),
                   nrow=2, ncol=2)
W <- c(0.2,0.8)
sim <- piv_sim(N,k,Mu,stdev,Sigma.p1,Sigma.p2,W)
res <- piv_MCMC(sim$y, k, nMC)
rel <- piv_rel(mcmc = res, nMC = nMC)
```

```
piv_plot(y=sim$y, mcmc=res, rel_est = rel, type="chains")
piv_plot(y=sim$y, mcmc=res, rel_est = rel,
         type="estimates_hist")
```

`piv_sel` *Pivotal Selection via Co-Association Matrix*

Description

Finding the pivots according to four different methods involving a co-association matrix C. This is an internal function launched by `piv_MCMC`.

Usage

```
piv_sel(Obj, k, gIndex, C, n, ZM, maxima, available_met)
```

Arguments

`Obj`	Numerical string for the allowed pivotal criterion.
`k`	The number of mixture components/groups.
`gIndex`	Clusters' allocation.
`C`	Co-association matrix.
`n`	Data sample size
`ZM`	Auxiliary matrix used for building `C`.
`maxima`	Initial assignment for MUS algorithm.
`available_met`	Available criteria methods (integer).

Value

`Cg`	The pivotal units.

`piv_sim` *Generate Data from a Gaussian Nested Mixture*

Description

Simulate N observations from a nested Gaussian mixture model with k pre-specified components.

Usage

```
piv_sim(N, k, Mu, stdev, Sigma.p1, Sigma.p2, W)
```

Arguments

N	Sample size, data dimension.
k	Number of mixture components.
Mu	Initial mean vector/matrix.
stdev	Initial standard deviations (for univariate mixtures).
Sigma.p1	Covariance matrix for the first mixture level (for bivariate mixtures only).
Sigma.p2	Covariance matrix for the second mixture level (for bivariate mixture only).
W	Mixture weights for the two levels,vector.

Value

y	Data values.

Examples

```
# Bivariate mixture simulation with three components

N  <- 2000
k  <- 3
M1 <- c(-45,8)
M2 <- c(45,.1)
M3 <- c(100,8)
Mu <- matrix(rbind(M1,M2,M3),c(k,2))
stdev    <- cbind(rep(1,k), rep(200,k))
Sigma.p1 <- matrix(c(stdev[1,1],0,0,stdev[1,1]),
nrow=2, ncol=2)
Sigma.p2 <- matrix(c(stdev[1,2],0,0,stdev[1,2]),
 nrow=2, ncol=2)
W   <- c(0.2,0.8)
sim <- piv_sim(N,k,Mu,stdev,Sigma.p1,Sigma.p2,W)
plot(sim$y, xlab="y[,1]", ylab="y[,2]")
```

Index