Package 'bclust'

September 15, 2015

Type Package

Title Bayesian Hierarchical Clustering Using Spike and Slab Models

Version 1.5

Date 2015-08-12

Author Vahid PARTOVI NIA and Anthony C. DAVISON

Maintainer Vahid PARTOVI NIA <vpartovinia@gmail.com>

Depends R (>= 3.2.1), stats, grDevices, graphics

Description Builds a dendrogram using log posterior as a natural distance defined by the model and meanwhile waits the clustering variables. It is also capable to computing equivalent Bayesian discrimination probabilities. The adopted method suites small sample large dimension setting. The model parameter estimation maybe difficult, depending on data structure and the chosen distribution family.

License GPL (>= 2)

URL http://bclust.r-forge.r-project.org/

Repository CRAN

Repository/R-Forge/Project bclust

Repository/R-Forge/Revision 41

Repository/R-Forge/DateTimeStamp 2015-08-27 17:06:01

Date/Publication 2015-09-15 00:33:08

NeedsCompilation yes

R topics documented:

`bclust` *Bayesian agglomerative clustering for high dimensional data with variable selection.*

Description

The function clusters data saved in a matrix using an additive linear model with disappearing random effects. The model has built-in spike-and-slab components which quantifies important variables for clustering and can be extracted using the `imp` function.

Usage

```
bclust(x,rep.id=1:nrow(x),effect.family="gaussian",
var.select=TRUE,transformed.par,labels=NULL)
```

Arguments

`x`	A numeric matrix, with clustering individuals in rows and variables in columns.
`rep.id`	A vector consisting of positive integer elements having the same length as the number of rows of `x`. This vector identifies replicates of a clustering type such that the total number of clustering types is `max(rep.id)`. If nothing is declared the function presupposes that the data are unreplicated, that is each row of `x` is a clustering type.
`effect.family`	Distribution family of the disappearing random components. The choices are "gaussian" or "alaplace" allowing Gaussian or asymmetric Laplace family, respectively.
`var.select`	A logical value, `TRUE` for fitting models that define spike-and-slab distribution in variable level and allows Bayesian variable selection.
`transformed.par`	The transformed model parameters in a vector. The length of the vector depends on the chosen model and the availability of variable selection. The log transformation is supposed to be applied for the variance parameters, the identity for the mean, and the logit for the proportions. The function loglikelihood can be used to estimate them from the data.
`labels`	A vector of strings referring to the labels of clustering types. The length of the vector should match to `max(rep.id)`. The first element corresponds to the label of the type having the smallest integer value in `rep.id`, the second element refers to the label of the type having the second smallest integer in `rep.id`, and so on.

Details

The function calls internal C functions depending on the chosen model. The C-stack of the system may overflow if you have a large dataset. You may need to adjust the stack before running R using your operation system command line. If you use Linux, open a console and type >ulimit -s unlimited and then run R in the same console. The Microsoft Windows users don't need to increase the stack size.

We assumed a Bayesian linear model for clustering being

$$y_{vctr} = \mu + \eta_{vct} + \delta_v \gamma_{vc} \theta_{vc} + \varepsilon_{vctr}$$

where y_{vctr} is the available data on variable v, cluster c, clustering type t, and replicate r; η_{vct} is the between-type error, θ_{vc} is the disappearing random component controlled by the Bernoulli variables δ_v with success probability q and γ_{vc} with success probability p; and ε_{vctr} is the between-replicate error. The types inside a cluster share the same θ_{vc}, but may arise with a different η_{vct}. For more details see Vahid Partovi Nia and Anthony C. Davison (2012)

Value

`data`	The data matrix, reordered according to `rep.id`.
`repno`	The number of replicates of the values of `rep.id`
`merge`	The merge matrix, in `hclust` object format.
`height`	A monotone vector referring to the height of the constructed tree.
`logposterior`	The log posterior for each merge.
`clust.number`	The number of clusters for each merge.
`cut`	The value of the height corresponding to the maximum of the log posterior in agglomerative path.
`transformed.par`	The transformed values of the model parameters. The log transformation is applied for the variance parameters, the identity for the mean, and the logit for the proportions.
`labels`	The labels associated to each clustering type.
`effect.family`	The distribution assigned to the disappearing random effect in the function arguments.
`var.select`	The variable selection chosen in the function arguments.

References

Vahid Partovi Nia and Anthony C. Davison (2012). High-Dimensional Bayesian Clustering with Variable Selection: The R Package bclust. Journal of Statistical Software, 47(5), 1-22. URL http://www.jstatsoft.org/v47/i05/

See Also

loglikelihood, meancss, imp.

Examples

```
data(gaelle)

# unreplicated clustering
gaelle.bclust<-bclust(x=gaelle,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
par(mfrow=c(2,1))
plot(as.dendrogram(gaelle.bclust))
abline(h=gaelle.bclust$cut)
plot(gaelle.bclust$clust.number,gaelle.bclust$logposterior,
xlab="Number of clusters",ylab="Log posterior",type="b")
abline(h=max(gaelle.bclust$logposterior))

#replicated clustering
gaelle.id<-rep(1:14,c(3,rep(4,13)))
# first 3 rows replication of ColWT
# 4 replications for the others

gaelle.lab<-c("ColWT","d172","d263","isa2",
"sex4","dpe2","mex1","sex3","pgm","sex1",
"WsWT","tpt","RLDWT","ke103")

gaelle.bclust<-bclust(gaelle,rep.id=gaelle.id,labels=gaelle.lab,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
plot(as.dendrogram(gaelle.bclust))
abline(h=gaelle.bclust$cut)
plot(gaelle.bclust$clust.number,gaelle.bclust$logposterior,
xlab="Number of clusters",ylab="Log posterior",type="b")
abline(h=max(gaelle.bclust$logposterior))
```

`bclustvs` *bclustvs (Bayesian CLUSTering with Variable Selection) is a class*

Description

The `bclustvs` object can be regarded as a Bayesian extension of the `hclust` object that additionally contains information of a Bayesian dendrogram. You can convert any `bclustvs` object to a `dendrogram` object using `as.dendrogram`.

Value

`data`	The data matrix, reordered according to `rep.id`.
`repno`	The number of replicates of the values of `rep.id`
`merge`	The merge matrix in `hclust` object format.
`height`	A monotone vector referring to the height of the constructed tree.
`logposterior`	The log posterior for each merge.
`clust.number`	The number of clusters in each merge.

cut	The value of the height corresponding to the maximum of the log posterior in agglomerative path.
transformed.par	The transformed values of the model parameters. The log transformation is applied for the variance parameters, the identity for the mean, and the logit for the proportions.
labels	The labels associated to each clustering type.
effect.family	The distribution assigned to the disappearing random effect.
var.select	Availability of variable selection chosen in the function arguments.

---

bdiscrim *discrimination using a Bayesian linear model*

Description

This function provides a discrimination method equivalent to the bclust clustering function.

Usage

```
bdiscrim(training, training.id,
training.labels = NULL, predict,
predict.label = rownames(predict)[1],
effect.family = "gaussian", var.select = TRUE,
transformed.par, priorprob = rep(1, max(training.id) + 1))
```

Arguments

training	A numeric matrix of the training set with observations in rows and variables in columns.
training.id	A vector consisting of positive integer elements having the same length as the number of rows of training. This vector identifies observations of a class such that the total number of classes is max(training.id).
training.labels	A vector of strings referring to the labels of each class. The length of the vector should match to max(training.id). The first element corresponds to the label of the type having the smallest integer value in training.id, the second element refers to the label of the type having the second smallest integer in training.id, and so on.
predict	A single discriminating type to be classified to one of the classes, with replications of the discriminating type in rows and variables in columns. The number of variables should be the same as in training. If it is not specified just the variable and variable-class importances will be calculated.
predict.label	A single string, the label of the discriminating type.

`effect.family` Distribution family of the disappearing random components of the model. The choices are "gaussian" or "alaplace" allowing Gaussian or asymmetric Laplace family, respectively.

`var.select` A logical value, `TRUE` for fitting models that define a spike-and-slab distribution in variable level and allows Bayesian variable selection.

`transformed.par`
The transformed model parameters in a vector. The length of the vector depends on the chosen model and the availability of variable selection. The log transformation is supposed to be applied for the variance parameters, the identity for the mean, and the logit for the proportions. The function loglikelihood can be used to estimate them from the data.

`priorprob` The prior probabilities for each class in a vector. The length of the vector should be the number of classes `max(training.id)` plus one (the prior probability that the predict set raises its own class). If nothing is specified a uniform discrete prior is considered.

Details

The function calls internal `C` functions depending on the chosen model. The C-stack of the system may overflow if you have a large dataset. You may need to adjust the stack before running `R` using your operational system command line.

We assumed a Bayesian linear model for classification being

$$y_{vctr} = \mu + \eta_{vct} + \delta_v \gamma_{vc} \theta_{vc} + \varepsilon_{vctr}$$

where y_{vctr} is the available data on variable v, cluster c, type t, and replicate r; η_{vct} is the between-type error, θ_{vc} is the disappearing random component controlled by the Bernoulli variables δ_v with success probability q and γ_{vc} with success probability p; and ε_{vctr} is the between-replicate error. The function computes the posterior probability that the `predict` data share the same θ_{vc}. This function also consideres that the `predict` data may arise its own class. For more details see Vahid Partovi Nia and Anthony C. Davison (2012)

Value

`probs` The posterior probabilities in a matrix with one row.

`var` The variable importances, each being a log Bayes factor.

`varclass` The variable-class importances, each being a log Bayes factor.

References

Vahid Partovi Nia and Anthony C. Davison (2012). High-Dimensional Bayesian Clustering with Variable Selection: The R Package bclust. Journal of Statistical Software, 47(5), 1-22. URL http://www.jstatsoft.org/v47/i05/

See Also

loglikelihood, bclust, profileplot.

Examples

```
data(gaelle)
gaelle.id<-rep(1:13,rep(4,13)) # all mutants have 4 replicates
gaelle.lab<-c("d172","d263","isa2",
"sex4","dpe2","mex1","sex3","pgm","sex1","WsWT",
"tpt","RLDWT","ke103")

gaelle.bdiscrim<-bdiscrim(gaelle[-(1:3),],
training.id=gaelle.id,training.labels=gaelle.lab,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68),
predict=gaelle[1:3,], predict.label="ColWT")
# classify ColWT to one of the types

par(mfrow=c(1,1)) #retreive graphic defaults
par(mar = c(5, 4, 4, 2) + 0.1) # leave some space for labels

ColWT.prob<-as.vector(gaelle.bdiscrim$probs)

# plots discrimination probabilities
bp <- barplot(ColWT.prob,ylim=c(0,1)) #plot bars
title("ColWT Discrimination")
text(bp, par("usr")[3]-0.05, srt = 90,adj=1,
labels = colnames(gaelle.bdiscrim$probs),
xpd = TRUE)
#plot variable labels

mtext(1, text = "Mutant", line = 4,cex=1.5)
# add x axis label
mtext(2, text = "Probability", line = 3,cex=1.2)
# add y axis labels
abline(h=1/length(ColWT.prob))
# draw plot a as a reference line prior probabilities line



# plots sorted discrimination probabilities
par(mfrow=c(1,1)) #retreive graphic defaults
par(mar = c(5, 4, 4, 2) + 0.1) # leave some space for labels
bp <- barplot(sort(ColWT.prob,decreasing=TRUE),ylim=c(0,1),
col=heat.colors(length(ColWT.prob))) #plot bars
text(bp, par("usr")[3]-0.05, srt = 90,adj=1,
labels = colnames(gaelle.bdiscrim$probs)
[order(ColWT.prob,decreasing=TRUE)],
xpd = TRUE) #plot variable labels
mtext(1, text = "Mutant", line = 4,cex=1.5)# add x axis label
mtext(2, text = "Probability", line = 3,cex=1.2)# add y axis labels
abline(h=1/length(ColWT.prob))


varclassimp<-gaelle.bdiscrim$varclass
#use thresholds to define blob colors
```

```
     blob<-matrix(0,nrow(varclassimp),ncol(varclassimp))
     blob[varclassimp<=0]<-0
     blob[varclassimp>0]<-1
     blob[varclassimp>1]<-2
     blob[varclassimp>3]<-3
     blob[varclassimp>5]<-4
#log bayes factor thresholding

varimp<-gaelle.bdiscrim$var
varcol<-rep(0,ncol(gaelle))
varcol[varimp>0]<-1


var.order<-order(gaelle.bdiscrim$var,decreasing=TRUE)
profileplot(x=gaelle[-(1:3),var.order],rep.id=gaelle.id,
labels=gaelle.lab,scale=10,blob.matrix=blob,ylab.mar=5,
xlab.mar=7)
#plot var class importance on profile plot using blobs
# and sort variables according to variable importance values



viplot(varimp=varimp, xlab=colnames(gaelle)[var.order],
xlab.mar=10,sort=TRUE,col=varcol[var.order])
#plot sorted variable importances
```

`ditplot` *dendrogram-image-teeth plot*

Description

This is a handy function to plot a `bclustvs` object. The function attaches a coloured horizontal dendrogram to the left side of an image plot with the optimal grouping highlighted by a teethplot on the right.

Usage

```
ditplot(x, xlab = colnames(x$data),
ylab = x$labels, xlab.cex = 1,
ylab.cex = 1, dendrogram.lwd = 1, dendrogram.size = 2,
xlab.mar = 3, ylab.mar = 3, image.col = rainbow(20),
horizbar.plot = FALSE,
horizbar.col = rev(c(heat.colors(5)[-4], "white")),
horizbar.distance = 4,varimp = rep(0, ncol(x$data)),
horizbar.size = 0.5, vertbar = NULL,
vertbar.col = rainbow(max(vertbar)),
teeth.size = 0.25, plot.width = 10)
```

Arguments

x	A bclustvs object.
xlab	A vector of strings elements. The labels for the clustering types automatically extracted from x.
ylab	A vector of strings. The variable labels automatically extracted from x.
xlab.cex	A positive value, the magnitude of the clustering type labels.
ylab.cex	A positive value, the magnitude of the variable labels.
dendrogram.lwd	A positive value, the thickness of lines used to plot the dendrogram.
dendrogram.size	A positive value, the size of the dendrogram plot.
xlab.mar	A positive value, the margin reserved to write variable labels.
ylab.mar	A positive value, the margin reserved to write type labels.
image.col	Colours used for the image plot.
horizbar.plot	A logical value, if TRUE a horizontal bar is plotted according to categorised varimp.
horizbar.col	Colours used for the horizontal bar.
horizbar.distance	A positive value, the distance between the horizbar and the image plot.
varimp	A numerical vector denoting the importance of variables. We propose to use the imp function to compute these values. If it is specified, the variables will be ordered respect to this vector.
horizbar.size	A positive value, the size of the horizontal bar.
vertbar	A positive integer vector that may be used to draw an additional vertical bar on the right of the teeth plot. This may be helpful to represent another optional grouping on the data.
vertbar.col	The colours used to plot the additional vertical bar.
teeth.size	A positive value, the size of the teeth plot.
plot.width	A positive value, the width of the whole plot. If the plot region is unbalanced in width and height, adjust this value.

Details

The varimp is assumed to be log Bayes factors and therefore categorised according to Kass and Raftery (1995) for a better visualisation.

References

Kass and Raftery (1995) Bayes Factors, Journal of the American Statistical Association, Vol. 90, pp. 773–795.

See Also

dptplot, teethplot,profileplot,viplot.

Examples

```
data(gaelle)

gaelle.bclust<-bclust(gaelle,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
ditplot(gaelle.bclust,varimp=imp(gaelle.bclust)$var,horizbar.plot=TRUE,
plot.width=5,horizbar.size=0.2,ylab.mar=4)
#unreplicated clustering

wildtype<-rep(1,55) #initiate a vector
wildtype[c(1:3,48:51,40:43)]<-2 #associate 2 to wildtypes
ditplot(gaelle.bclust,varimp=imp(gaelle.bclust)$var,horizbar.plot=TRUE,
plot.width=5,horizbar.size=0.2,vertbar=wildtype,
vertbar.col=c("white","violet"),ylab.mar=4)
#mark wildtype plants using violet

gaelle.id<-rep(1:14,c(3,rep(4,13)))
# first 3 rows replication of ColWT, 4 for the rest
gaelle.lab<-c("ColWT","d172","d263","isa2",
"sex4","dpe2","mex1","sex3","pgm","sex1","WsWT","tpt","RLDWT","ke103")
gaelle.bclust<-bclust(gaelle,rep.id=gaelle.id,
labels=gaelle.lab,transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
ditplot(gaelle.bclust,varimp=imp(gaelle.bclust)$var,horizbar.plot=TRUE)
#replicated clustering
```

---

dptplot *dendrogram-profile-teeth plot*

Description

This is a plot function suitable for visualisation of a `bclustvs` object when the `bclust` function is used on replicated data, but can be applied for unreplicated data too. The function attaches a coloured horizontal dendrogram to the left side of a profile plot with the optimal grouping highlighted by a teethplot on the right.

Usage

```
dptplot(x, scale = 1, xlab = colnames(x$data), ylab = x$labels,
xlab.cex = 1, ylab.cex = 1, dendrogram.lwd = 1,
dendrogram.size = 2, xlab.mar = 3, ylab.mar = 3,
horizbar.plot = FALSE,
horizbar.col = rev(c(heat.colors(5)[-4], "white")),
horizbar.distance = 4,  varimp = rep(0, ncol(x$data)),
horizbar.size = 0.5, vertbar = NULL,
vertbar.col = rainbow(max(vertbar)),
teeth.size = 0.25, plot.width = 10)
```

Arguments

x	A bclustvs object.
scale	A positive value. If the profile plots are too noisy or too flat, adjust the scaling factor.
xlab	A string vector, the labels for the clustering types.
ylab	A string vector, the variable labels.
xlab.cex	A positive value, the magnitude of the type labels.
ylab.cex	A positive value, the magnitude of the variable labels.
dendrogram.lwd	A positive value, the thickness of lines used to plot the dendrogram.
dendrogram.size	A positive value, the size of the dendrogram plot.
xlab.mar	A positive value, the margin reserved to write variable labels.
ylab.mar	A positive value, the margin reserved to write type labels.
horizbar.plot	A logical value. If TRUE a horizontal bar is plotted according to the varimp.
horizbar.col	Colours of the horizontal bar.
horizbar.distance	A positive value, the distance between the horizontal bar and the profile plot.
varimp	A numerical vector denoting the importance of variables. We propose to use imp function to get the values. If it is specified, the variables will be ordered respect to this vector.
horizbar.size	A positive value, the size of the horizontal bar.
vertbar	A positive integer vector that may be used to draw an additional vertical bar on the right of the teethplot. This may be helpful to represent another optional grouping on the data.
vertbar.col	The colours used to plot the additional vertical bar.
teeth.size	A positive integer, the size of the teeth plot.
plot.width	A positive integer, the width of the whole plot. If the plot region is unbalanced in width and height, adjust this value.

Details

The varimp is assumed to be the log Bayes factor and therefore categorised according to Kass and Raftery (1995) for a better visualisation. If varimp is not result of the imp function, keep horizbar.plot=FALSE.

References

Kass and Raftery (1995) Bayes Factors, Journal of the American Statistical Association, Vol. 90, pp. 773–795.

See Also

ditplot, profileplot, teethplot, viplot.

Examples

```
data(gaelle)

gaelle.id<-rep(1:14,c(3,rep(4,13)))
# first 3 rows replication of ColWT, 4 for the rest

gaelle.lab<-c("ColWT","d172","d263","isa2",
"sex4","dpe2","mex1","sex3","pgm","sex1","WsWT","tpt","RLDWT","ke103")
gaelle.bclust<-bclust(gaelle,rep.id=gaelle.id,labels=gaelle.lab,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
dptplot(gaelle.bclust,scale=5,varimp=imp(gaelle.bclust)$var,
horizbar.plot=TRUE)
#replicated clustering


gaelle.bclust<-bclust(gaelle,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
dptplot(gaelle.bclust,scale=10,varimp=imp(gaelle.bclust)$var,
horizbar.plot=TRUE,plot.width=5,horizbar.size=0.2,ylab.mar=4)
#unreplicated clustering

wildtype<-rep(1,55) #initiate a vector
wildtype[c(1:3,48:51,40:43)]<-2 #associate 2 to wildtypes
dptplot(gaelle.bclust,scale=10,varimp=imp(gaelle.bclust)$var,
horizbar.plot=TRUE,plot.width=5,horizbar.size=0.2,vertbar=wildtype,
vertbar.col=c("white","violet"),ylab.mar=4)
```

---

gaelle *Messerli et. al. metabolomic data*

Description

The dataset consists of replicates of 14 mustants to study metabolite pathways of genetically modified samples of *Arabidopsis thaliana*. Values of 43 metabolites are measured for each sample which are supposed to monitor their genetic changes. The data involve two mutants defective in starch biosynthesis, *pgm* and *isa2*; four defective in starch degradation *sex1*, *sex4*, *mex1*, and *dpe2*; a mutant for comparison that accumulates starch as a pleitropic effect, *tpt*; four uncharacterised mutants, *deg172*, *deg263*, *ke103*, and *sex3*; and three wild type plants, *WsWT*, *RLDWT*, and *ColWT*. There are four replicates of all samples except the last for which there are three http://www.plantphysiol.org/cgi/content/abstract/143/4/1484.

Usage

```
data(gaelle)
```

Format

Matrix with 55 observations measured on 43 variables.

References

Messerli, G., Partovi Nia, V., Trevisan, M., Kolbe, A., Schauer, N., Geigen-berger, P., Chen, J., Davison, A. C., Fernie, A. R. and Zeeman, S. C. (2007) Rapid classification of phenotypic mutants of Arabidopsis via metabolite fingerprinting. Plant Physiology 143, 1481-1492.

Examples

```
data(gaelle)
heatmap(gaelle)
```

`imp` *calculates variable and variable-cluster importances*

Description

The function computes the log Bayes factors for the hypothesis H0: the variable or the variable-cluster combination is useful for clustering against H1: the variable or the variable-cluster combination is useless. The Bayes factors are computed for the optimal allocation found by the `bclust` function.

Usage

```
imp(x)
```

Arguments

`x`	A `bclustvs` object.

Value

`var`	A vector being the log Bayes factor of $\delta_v = 1$ against $\delta_v = 0$, see bclust for details.
`varclust`	A vector being the log Bayes factor of $\gamma_{vc} = 1$ against $\gamma_{vc} = 0$, see bclust for details.
`repno`	The number of replicates producing each row of `varclust`.
`labels`	The vector of variable labels extracted from the `bclustvs` object.
`order`	The order of `var` useful to sort `var`, `varclust`, and `labels`.

See Also

bclust.

Examples

```
data(gaelle)

gaelle.id<-rep(1:14,c(3,rep(4,13)))
# first 3 rows replication of ColWT, 4 for the rest

gaelle.bclust<-bclust(gaelle,rep.id=gaelle.id,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68),
var.select=TRUE)

gaelle.imp<-imp(gaelle.bclust)

#plot the variable importances
par(mfrow=c(1,1)) #retreive graphic defaults

mycolor<-gaelle.imp$var
mycolor<-c()
mycolor[gaelle.imp$var>0]<-"black"
mycolor[gaelle.imp$var<=0]<-"white"

viplot(var=gaelle.imp$var,xlab=gaelle.imp$labels,col=mycolor)
#plot important variables with balck

viplot(var=gaelle.imp$var,xlab=gaelle.imp$labels,
sort=TRUE,col=heat.colors(length(gaelle.imp$var)),
xlab.mar=10,ylab.mar=4)
mtext(1, text = "Metabolite", line = 7,cex=1.5)# add x axis label
mtext(2, text = "Log Bayes Factor", line = 3,cex=1.2)# add y axis labels
#sort importnaces and use heat colors add some labels to the x and y axes
```

`loglikelihood` *computes the model log likelihood useful for estimation of the transformed.par*

Description

The function is useful for deriving the maximum likelihood estimates of the model parameters.

Usage

```
loglikelihood(x.mean,x.css,repno,transformed.par,
effect.family="gaussian",var.select=TRUE)
```

Arguments

`x.mean`	The mean matrix of the clustering types from the meancss function.
`x.css`	The corrected sum of squares matrix of the clustering types from the meancss function.

`repno` The vector containing the number of replications of each clustering type corresponding to the each row of `x.mean` and `x.css`, from the meancss function.

`transformed.par`
The vector of transformed model parameters that the data likelihood will be evaluated at. The transformation is the log for the variance parameters, the identity for the mean, and the logit for the proportions. The length of the vector depends on the chosen `effect.family` and `var.select`.

`effect.family` Distribution family of the disappearing random components. Choices are "gaussian" or "alaplace" allowing Gaussian or asymmetric Laplace family, respectively.

`var.select` A logical value, `TRUE` for fitting models that define spike-and-slab in variable level, thus allowing Bayesian variable selection.

Details

Sometimes estimation of the model parameters is difficult, always check the convergence of the optimisation algorithm. The asymmetric Laplace model, `effect.family="alaplace"`, is often more difficult to optimise than `effect.family="gaussian"`.

If data are standardised (having general mean zero and general variance one) the log likelihood function is usually maximised over values between -5 and 5.

The `transformed.par` is a vector of transformed model parameters having length 5 up to 7 depending on the chosen model.

The `transformed.par` is $(\log\sigma^2, \log\sigma^2_\eta, \log\sigma^2_\theta, \mu, \text{logit}p, \text{logit}q)$ a vector of length 6 when using `effect.family = "gaussian"` and `var.select=TRUE`,

and is $(\log\sigma^2, \log\sigma^2_\eta, \log\sigma^2_{\theta_L}, \log\sigma^2_{\theta_R}, \mu, \text{logit}p, \text{logit}q)$ a vector of length 7

for `effect.family="alaplace"` and `var.select=TRUE`.

When `var.select=FALSE` the q parameter is dropped, yielding a vector of length 5 for

`effect.family="gaussian"` and a vector of length 6

for `effect.family="alaplace"`.

We assumed a Bayesian linear model being

$$y_{vctr} = \mu + \eta_{vct} + \delta_v\gamma_{vc}\theta_{vc} + \varepsilon_{vctr}$$

where y_{vctr} is the available data on variable v, cluster(or class) c, type t, and replicate r; η_{vct} is the between-type error, θ_{vc} is the disappearing random component controlled by the Bernoulli variables δ_v with success probability q and γ_{vc} with success probability p; and ε_{vctr} is the between-replicate error. The types inside a cluster (or class) share the same θ_{vc}, but may arise with a different η_{vct}.

The model parameters has natural interpretations, σ^2 is the between replicate error variance; σ^2_η is the variance of between-type error; σ^2_θ is the variance of the disappearing random component which is decomposed to $\sigma^2_{\theta_L}$, $\sigma^2_{\theta_R}$ the left and the right tail variances if the model is asymmetric Laplace; μ is the general level; p is the proportion of active variable-cluster (or variable-class) combinations, and q is the proportion of the active variables. For more details see Vahid Partovi Nia and Anthony C. Davison (2012)

References

Vahid Partovi Nia and Anthony C. Davison (2012). High-Dimensional Bayesian Clustering with Variable Selection: The R Package bclust. Journal of Statistical Software, 47(5), 1-22. URL http://www.jstatsoft.org/v47/i05/

See Also

bclust,bdiscrim, meancss.

Examples

```
data(gaelle)
gaelle.id<-rep(1:14,c(3,rep(4,13)))
# first 3 rows replication of ColWT, 4 for the rest

mc.gaelle<-meancss(gaelle,gaelle.id)

optimfunc<-function(theta)
{
-loglikelihood(x.mean=mc.gaelle$mean,x.css=mc.gaelle$css,
repno=mc.gaelle$repno,transformed.par=theta)#compute - log likelihood
}

transpar<-optim(rep(0,6),optimfunc,method="BFGS")$par
#gives argmin(-loglikelihood)
#put a vector of correct length for the evaluation of the likelihood

plot(bclust(gaelle,transformed.par=transpar))
```

`meancss` *computes statistics necessary for the evaluation of the log likelihood*

Description

The function is useful for deriving arguments of the loglikelihood function.

Usage

```
meancss(x,rep.id=1:nrow(x))
```

Arguments

`x`	The data matrix, subjects in rows, variables in columns.
`rep.id`	A vector of positive integers refering to replication of the types. The same integer is associated to the the replicates of the same type. Not specifying this vector preproposes that the data are unreplicated.

Details

This function facilitates the usage of the `loglikelihood` function.

Value

`mean`	The mean of types.
`css`	The corrected sum of squares of types.
`repno`	The vector containing the number of replications of types according to `rep.id`.

See Also

loglikelihood.

Examples

```
data(gaelle)
gaelle.id<-rep(1:14,c(3,rep(4,13)))
# first 3 rows replication of ColWT, 4 for the rest
mc.gaelle<-meancss(gaelle,gaelle.id)
loglikelihood(x.mean=mc.gaelle$mean,x.css=mc.gaelle$css,
repno=mc.gaelle$repno,transformed.par=rep(0,6))
# evalutes likelihood at rep(0,6)
```

`profileplot` *a plot useful to visualise replicated data*

Description

The profile plot is a flexible function creating profiles of replicated data with many useful options. The resulting plot can be attached to other plots like a horizontal dendrogram plot or a teethplot.

Usage

```
profileplot(x, rep.id, labels = NULL, scale = 1,
col.names = colnames(x), plot.order = 1:max(rep.id),
xlab.mar = 5, ylab.mar = 5, xlab.cex = 0.8,
ylab.cex = 1, profile.col = rep(1, max(rep.id)),
blob.matrix = matrix(0, ncol = ncol(x), nrow = max(rep.id)),
blob.col = rev(heat.colors(max(blob.matrix))), blob.cex = 1.5)
```

Arguments

`x`	The data matrix with subjects in rows and variables in columns.
`rep.id`	A positive integer vector referring to replication of a subject such that the total number of subjects is `max(rep.id)`.

labels	The labels of each subject. The first element corresponds to the subject that takes value 1 in rep.id, the second element is the one that takes value 2 in rep.id and so on.
scale	A positive value. If the profile plots are noisy or flat, adjust the scaling factor.
col.names	The data column labels (variable labels) in a string vector.
plot.order	A positive integer vector, the order that subjects should be plotted. This is useful if you like to attach this plot to a dendrogram or a teeth plot.
xlab.mar	A positive value, the margin reserved for the column (variable) labels.
ylab.mar	A positive value, the margin reserved for the subject labels.
xlab.cex	A positive value, the magnitude of the variable labels.
ylab.cex	A positive value, the magnitude of the subject labels.
profile.col	Colours that you may used to plot each profile.
blob.matrix	An integer matrix denoting where to plot a colour blob and which colour should be used in which location. The number of rows of this matrix should be max(rep.id) and the number of columns should be ncol(x). The value 0 plots no blobs, value 1 refers to plotting a blob with a colour specified in blob.col.
blob.col	The colours that is used for each value of blob.matrix
blob.cex	The magnitude of blobs.

Details

Some of the options may be useless for just a profile plot, but all of them are beneficial for a good visual representation of data when the plot is attached to a dendrogram or a teethplot.

See Also

teethplot, ditplot, dptplot.

Examples

```
data(gaelle)
#take a subset of gaelle data
subgaelle<-gaelle[1:11,]

#use thresholds to define blob colors
blob<-matrix(0,nrow(subgaelle),ncol(subgaelle))
blob[abs(subgaelle)<=0.74]<-0
blob[abs(subgaelle)>0.74]<-1
blob[abs(subgaelle)>0.94]<-2
blob[abs(subgaelle)>1.28]<-3
profileplot(subgaelle,1:nrow(subgaelle),scale=10,blob.matrix=blob)

#make a profile plot with colored blobs
####### attach a profileplot to a teeth plot
subgaelle.bclust<-bclust(subgaelle,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
# divide plot space into two unequal parts
```

```
layout(matrix(c(1,2),1,2,byrow=TRUE), c(9,1),10, respect=TRUE)
# associate a color to each subject
subgaelle.color<-c(rep("blue",3),rep("green",4),rep("magenta",4))
# find appropriate order to plot subjects
subgaelle.order<-order.dendrogram(as.dendrogram(subgaelle.bclust))
#leave some space for labels
x.mar<-7
y.mar<-5
profileplot(subgaelle,rep.id=1:nrow(subgaelle),scale=10,
profile.col=subgaelle.color,plot.order=subgaelle.order,
xlab.mar=x.mar,ylab.mar=y.mar)
par(mar=c(x.mar,0,0,0))
teethplot(subgaelle.bclust)

####### nowattach it to a dendrogram plot
layout(matrix(c(2,1),1,2,byrow=TRUE), c(2,8),10, respect=TRUE)
profileplot(subgaelle,rep.id=1:nrow(subgaelle),scale=10,
profile.col=subgaelle.color,plot.order=subgaelle.order,
xlab.mar=x.mar,ylab.mar=y.mar)
par(mar=c(x.mar,0,0,0))
plot(as.dendrogram(subgaelle.bclust),horiz=TRUE,yaxs="i",
axes=FALSE,leaflab="none")
abline(v=subgaelle.bclust$cut)
```

`teethplot` *produces teeth plot useful for demonstating a grouping on clustered subjects*

Description

The function alone is useless but can be attached to a horizontal dendrogram, a profile plot, or an image plot to show a specified partitioning.

Usage

```
teethplot(x, teeth.space = 0.25, teeth.lwd = 1)
```

Arguments

`x`	A `bclustvs` object.
`teeth.space`	The space between two teeth, a value between 0 and 0.25.
`teeth.lwd`	The thickness of the lines used to draw the teeth.

Details

The teeth plot for the moment shows the grouping vertically.

See Also

profileplot, ditplot, dptplot.

Examples

```
data(gaelle)
gaelle.id<-rep(1:14,c(3,rep(4,13)))
# first 3 rows replication of ColWT, 4 for the rest

gaelle.lab<-c("ColWT","d172","d263","isa2","sex4","dpe2","mex1",
"sex3","pgm","sex1","WsWT","tpt","RLDWT","ke103")
gaelle.bclust<-bclust(gaelle,rep.id=gaelle.id,labels=gaelle.lab,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68),var.select=TRUE)
#start plotting
layout(matrix(c(1,2),1,2,byrow=TRUE), c(9,1),10, respect=TRUE)
# divide plot space into two unequal parts
par(mar=c(0,0,0,2))
# preserve some space for labels in dendrogram plot
plot(as.dendrogram(gaelle.bclust),
horiz=TRUE,yaxs="i") #plot the dendrogram
abline(v=gaelle.bclust$cut)
#show the optimal allocation by a line on the dendrogram
par(mar=c(0,0,0,0)) # we need no space for teeth plot
teethplot(gaelle.bclust) #show the optimal allocation using teeth plot
```

---

viplot *variable importance plot*

Description

This function plots varible importances using a barplot.

Usage

```
viplot(varimp, xlab, xlab.mar = 5, ylab.mar = 4, xlab.srt = 90,
xlab.cex = 1, sort = FALSE, ...)
```

Arguments

varimp	A numeric vector, variable importances.
xlab	A vector of strings. Labels to be plotted on x-axis.
xlab.mar	A positive value. The margin reserved for x-axis labels.
ylab.mar	A positive value, The margin reserved for y-axis labels.
xlab.srt	A numeric value, amount of rotation of the x-axis labels.
xlab.cex	A positive value, magnitude of x-axis labels.
sort	A logical value, TRUE if sorted variables should be plotted.
...	The barplot options.

Details

It is not strightforward to rotate x labels of a `barplot`. This function does it easily.

See Also

imp, dptplot, ditplot.

Examples

```
data(gaelle)
gaelle.bclust<-bclust(gaelle,
transformed.par=c(-1.84,-0.99,1.63,0.08,-0.16,-1.68))
gaelle.imp<-imp(gaelle.bclust)
viplot(varimp=gaelle.imp$var)
# solid plot

viplot(varimp=gaelle.imp$var,xlab=imp(gaelle.bclust)$labels,
sort=TRUE,col=heat.colors(length(gaelle.imp$var)))
# sorted plot with heat colors and labels
```

Index