Package 'HDclust'

October 12, 2022

Type Package

Title Clustering High Dimensional Data with Hidden Markov Model on Variable Blocks

Version 1.0.3

Date 2019-04-05

Description Clustering of high dimensional data with Hidden Markov Model on Variable Blocks (HMM-VB) fitted via Baum-Welch algorithm. Clustering is performed by the Modal Baum-Welch algorithm (MBW), which finds modes of the density function. Lin Lin and Jia Li (2017) <http://jmlr.org/papers/v18/16-342.html>.

License GPL (>= 2)

Imports Rcpp (>= 0.12.16), RcppProgress (>= 0.1), Rtsne (>= 0.11.0)

Depends methods

LinkingTo Rcpp, RcppProgress

Collate 'AllClass.R' 'AllGeneric.R' 'AllMethod.R' 'clustControl.R' 'trainControl.R' 'hmmvbClust.R' 'hmmvbTrain.R' 'vbSearchControl.R' 'package-HDclust.R' 'RcppExports.R' 'sim3.R' 'sim2.R' 'hmmvbBIC.R' 'hmmvbFindModes.R' 'clustModes.R'

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation yes

Encoding UTF-8

Author Yevhen Tupikov [aut],
Lin Lin [aut],
Lixiang Zhang [aut],
Jia Li [aut, cre]

Maintainer Jia Li <jiali@psu.edu>

Repository CRAN

Date/Publication 2019-04-11 21:57:32 UTC

R topics documented:

`HDclust-package` *Clustering high dimensional data with Hidden Markov Model on Variable Blocks*

Description

Clustering of high dimensional data with Hidden Markov Model on Variable Blocks (HMM-VB) fitted via Baum-Welch algorithm. Clustering is performed by the Modal Baum-Welch algorithm (MBW), which finds modes of the density function.

Details

For a quick introduction to **HDclust** see the vignette `vignette("HDclust")`.

Author(s)

Jia Li, Lin Lin and Yevhen Tupikov.

Maintainer: Yevhen Tupikov <yzt116@psu.edu>

References

Lin Lin and Jia Li, "Clustering with hidden Markov model on variable blocks," **Journal of Machine Learning Research**, 18(110):1-49, 2017.

See Also

`hmmvbTrain`, `hmmvbClust`

Examples

```
data("sim3")
set.seed(12345)
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
clust <- hmmvbClust(sim3[,1:40], model=hmmvb)
show(clust)
```

---

clustControl *Parameters for MBM clustering algorithm.*

Description

This function creates a list with parameters for Modal Baum-Welch (MBW) clustering algorithm used as an argument for `hmmvbClust`.

Usage

```
clustControl(minSize = 1, modeTh = 0.01, useL1norm = FALSE,
  getlikelh = FALSE)
```

Arguments

`minSize`	Minimum cluster size. Clusters that contain the number of data points smaller than `minSize` are merged to the closest big cluster.
`modeTh`	Distance parameter that controls mode merging. Larger values promote merging of different clusters.
`useL1norm`	A logical value indicating whether or not L1 norm will be used to calculate the distance.
`getlikelh`	A logical value indicating whether or not to calculate the loglikelihood for every data point.

Value

The named list with parameters.

See Also

hmmvbTrain

Examples

```
# avoid clusters of size < 60
Vb <- vb(1, dim=4, numst=2)
set.seed(12345)
hmmvb <- hmmvbTrain(iris[,1:4], VbStructure=Vb)
clust <- hmmvbClust(iris[,1:4], model=hmmvb, control=clustControl(minSize=60))
show(clust)
```

---

clustModes *Hierarchical clustering of density modes*

Description

This function performs hierarchical clustering of density modes found by hmmvbFindModes().

Usage

```
clustModes(modes, cutree.args, hclust.args = NULL, dist.args = NULL)
```

Arguments

modes	An object of class 'HMMVBclust' returned by hmmvbFindModes().
cutree.args	A list with arguments to cutree method from package stats.
hclust.args	A list with arguments to hclust method from package stats. It is used to specify a linkage method for hierarchical clustering of density modes. Complete linkage is used by default. See hclust for details.
dist.args	A list with arguments to dist method from package stats. It is used to specify a distance metric for the density modes. Euclidian distance is used by default. See dist for details.

Value

An object of class 'HMMVBclust' with new cluster labels and cluster sizes. Note that coordinates of modes after merging are not calculated and clustParam field is empty.

See Also

hmmvbClust, hmmvbFindModes

Examples

```
Vb <- vb(1, dim=4, numst=2)
set.seed(12345)
hmmvb <- hmmvbTrain(unique(iris[,1:4]), VbStructure=Vb)
modes <- hmmvbFindModes(unique(iris[,1:4]), model=hmmvb)

# default mode clustering
merged <- clustModes(modes, cutree.args=list(h=1.0))

# mode clustering using Manhattan distance
merged <- clustModes(modes, dist.args=list(method="manhattan"), cutree.args=list(h=1.0))

# mode clustering using single linkage
merged <- clustModes(modes, hclust.args=list(method="single"), cutree.args=list(h=1.0))
```

---

getBdim *Accessor for 'bdim' slot*

Description

This function outputs dimensionality of blocks of variable block structure.

Usage

```
getBdim(object)

## S4 method for signature 'VB'
getBdim(object)

## S4 method for signature 'HMMVB'
getBdim(object)
```

Arguments

object	Object of class "VB" or "HMMVB".

Examples

```
# accessing bdim in instance of class VB
Vb <- vb(2, dim=10, bdim=c(4,6), numst=c(3,11), varorder=list(c(1:4),c(5:10)))
getBdim(Vb)

# accessing bdim in instance of class HMMVB
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
getBdim(hmmvb)
```

getBIC *Accessor for 'BIC' slot.*

Description

This function outputs BIC for a trained HMM-VB model or a vector with BIC values calculated in model selection.

Usage

```
getBIC(object)

## S4 method for signature 'HMMVB'
getBIC(object)

## S4 method for signature 'HMMVBBIC'
getBIC(object)
```

Arguments

object	Object of class "HMMVB" or "HMMVBBIC".

getClsid *Accessor for 'clsid' slot.*

Description

This function outputs the cluster labels for the object of class HMMVBclust.

Usage

```
getClsid(object)
```

Arguments

object	Object of class "HMMVBclust".

`getClustParam` *Accessor for 'clustParam' slot.*

Description

This function outputs clusterPar for the object of class HMMVBclust.

Usage

```
getClustParam(object)
```

Arguments

`object`	Object of class "HMMVBclust".

`getDiagCov` *Accessor for 'diagCov' slot.*

Description

This function outputs diagCov logical indicator of diagonal covariance matrices for HMM-VB model.

Usage

```
getDiagCov(object)
```

Arguments

`object`	Object of class "HMMVB".

`getDim` *Accessor for 'dim' slot*

Description

This function outputs dimensionality.

Usage

```
getDim(object)

## S4 method for signature 'VB'
getDim(object)

## S4 method for signature 'HMM'
getDim(object)

## S4 method for signature 'HMMVB'
getDim(object)
```

Arguments

object	Object of class "VB", "HMM" or "HMMVB".

Examples

```
# accessing dim in instance of class VB
Vb <- vb(nb=2, dim=10, bdim=c(4,6), numst=c(3,11), varorder=list(c(1:4),c(5:10)))
getDim(Vb)

# accessing dim in instance of class HMM
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
getDim(getHmmChain(hmmvb)[[1]])

# accessing dim in instance of class HMMVB
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
getDim(hmmvb)
```

---

getHmmChain *Accessor for 'HmmChain' slot.*

Description

This function outputs a list with trained HMMs.

Usage

```
getHmmChain(object)
```

Arguments

object	Object of class "HMMVB".

---

getHmmParam *Accessor for parameters of HMM*

Description

This function outputs a list with means, covariance matrices, inverse covarince matrices and logarithms of the determinants of the covariance matrices for all states of the HMM.

Usage

```
getHmmParam(object)
```

Arguments

object	Object of class "HMM".

---

getLoglikehd *Accessor for 'Loglikehd' slot.*

Description

This function outputs Loglikelihood for each data point in a trained HMM-VB model or Loglikelihood for a new dataset in a HMM-VB model.

Usage

```
getLoglikehd(object)

## S4 method for signature 'HMMVB'
getLoglikehd(object)

## S4 method for signature 'HMMVBBIC'
getLoglikehd(object)

## S4 method for signature 'HMMVBclust'
getLoglikehd(object)
```

Arguments

object	Object of class "HMMVB", "HMMVBBIC" "HMMVBclust".

getNb *Accessor for 'nb' slot*

Description

This function outputs number of blocks of the variable block structure.

Usage

```
getNb(object)

## S4 method for signature 'VB'
getNb(object)

## S4 method for signature 'HMMVB'
getNb(object)
```

Arguments

object	Object of class "VB" or "HMMVB".

Examples

```
# accessing nb in instance of class VB
Vb <- vb(2, dim=10, bdim=c(4,6), numst=c(3,11), varorder=list(c(1:4),c(5:10)))
getNb(Vb)

# accessing nb in instance of class HMMVB
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
getNb(hmmvb)
```

getNumst *Accessor for 'numst' slot*

Description

This function outputs the number of states for each variable block in the variable block structure, the number of states of the HMM, or the number of states for each variable block of the HMM-VB.

Usage

```
getNumst(object)

## S4 method for signature 'VB'
getNumst(object)

## S4 method for signature 'HMM'
getNumst(object)

## S4 method for signature 'HMMVB'
getNumst(object)
```

Arguments

object	Object of class "VB", "HMM" or "HMMVB".

Examples

```
# accessing numst in instance of class VB
Vb <- vb(2, dim=10, bdim=c(4,6), numst=c(3,11), varorder=list(c(1:4),c(5:10)))
getNumst(Vb)

# accessing getNumst in instance of class HMM
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
getNumst(getHmmChain(hmmvb)[[1]])

# accessing numst in instance of class HMMVB
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
getNumst(hmmvb)
```

---

getOptHMMVB *Accessor for 'optHMMVB' slot.*

Description

This function outputs the optimal HMM-VB found via BIC model selection.

Usage

```
getOptHMMVB(object)
```

Arguments

object	Object of class "HMMVBBIC".

---

getPrenumst *Accessor for 'prenumst' slot*

Description

This function outputs the number of states in the HMM for the preceding block of HMM-VB.

Usage

```
getPrenumst(object)
```

Arguments

object	Object of class "HMM".

---

getSize *Accessor for 'size' slot.*

Description

This function outputs the number of points in each cluster for the object of class HMMVBclust.

Usage

```
getSize(object)
```

Arguments

object	Object of class "HMMVBclust".

getVarorder *Accessor for 'varorder' slot*

Description

This function outputs the ordering of the variable blocks.

Usage

```
getVarorder(object)

## S4 method for signature 'VB'
getVarorder(object)

## S4 method for signature 'HMMVB'
getVarorder(object)
```

Arguments

object	Object of class "VB" or "HMMVB".

Examples

```
# accessing varorder in instance of class VB
Vb <- vb(2, dim=10, bdim=c(4,6), numst=c(3,11), varorder=list(c(1:4),c(5:10)))
getVarorder(Vb)

# accessing varorder in instance of class HMMVB
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
getVarorder(hmmvb)
```

getVb *Accessor for 'VbStructure' slot.*

Description

This function outputs the variable block structure in the HMM-VB.

Usage

```
getVb(object)
```

Arguments

object	Object of class "HMMVB".

HMM-class *Class "HMM" to represent parameters associated with a variable block in the HMM-VB*

Description

An S4 class to represent the model parameters associated with one variable block in the HMM-VB. For brevity, we call this part of HMM-VB, specific to a particular variable block, an "HMM" for the block. New instances of the class are created by hmmvbTrain.

Methods

- **show** signature(object = "HMM") : show parameters of the HMM object.
- **getPrenumst** signature(object = "HMM") : accessor for 'prenumst' slot.
- **getHmmParam** signature(object = "HMM") : accessor for parameters of the HMM object. This function outputs a list with means, covariance matrices, inverse covarince matrices and logarithms of the determinants of the covariance matrices for all states of the HMM.

Slots

dim Dimensionality of the data in HMM.

numst An integer vector specifying the number of HMM states.

prenumst An integer vector specifying the number of states of previous variable block HMM.

a00 Probabilities of HMM states.

a Transition probability matrix from states in the previous variable block to the states in the current one.

mean A numerical matrix with state means. *k*th row corresponds to the *k*th state.

sigma A list containing the covariance matrices of states.

sigmaInv A list containing the inverse covariance matrices of states.

sigmaDetLog A vector with $log(|sigma|)$ for each state.

HMMVB-class *Class "HMMVB" to represent a Hidden Markov Model on Variable Blocks (HMM-VB).*

Description

An S4 class to represent a Hidden Markov Model on Variable Blocks (HMM-VB). New instances of the class are created by `hmmvbTrain`.

Methods

- **show** signature(object = "HMMVB") : show parameters of the HMM-VB.
- **getHmmChain** signature(object = "HMMVB") : accessor for 'HmmChain' slot.
- **getDiagCov** signature(object = "HMMVB") : accessor for 'diagCov' slot.
- **getBIC** signature(object = "HMMVB") : accessor for 'BIC' slot.
- **getVb** signature(object = "HMMVB") : accessor for 'VbStructure' slot.

Slots

`VbStructure` An object of class 'VB' that contains the variable block structure.

`HmmChain` A list of objects of class 'HMM' with trained Hidden Markov Models for each variable block.

`diagCov` A logical value indicating whether or not covariance matrices for mixture models are diagonal.

`Loglikehd` Loglikelihood value for each data point.

`BIC` BIC value for provided variable block structure or optimal BIC value for found variable block structure.

hmmvbBIC *BIC for HMM-VB*

Description

This function finds an optimal number of mixture components (states) for HMM-VB using the Bayesian Information Criterion (BIC). The variable block structure is provided as input and then BIC is estimated for HMM-VB with different configurations of states for the variable blocks.

Usage

```
hmmvbBIC(data, VbStructure, configList = NULL, numst = 1:10,
  trControl = trainControl(), nthread = 1)
```

Arguments

data	A numeric vector, matrix, or data frame of observations. Categorical values are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
VbStructure	An object of class 'VB'. Variable block structure stored in VbStructure is used to train HMM-VB model. numst parameter of the variable block structure is ignored.
configList	A list of integer vectors specifying number of states in each variable block for which BIC is to be calculated.
numst	An integer vector specifying the numbers of mixture components (states) in each variable block for which BIC is to be calculated. Number of states is the same for all variable blocks. The argument is ignored if configList argument is provided.
trControl	A list of control parameters for HMM-VB training algorithm. The defaults are set by the call hmmvbTrainControl().
nthread	An integer specifying the number of threads used in searching and training routines.

Value

A named list with estimated BIC values and the number of states or state configurations for which BIC was calculated.

See Also

VB, vb, trainControl

Examples

```
# Default search for the optimal number of states for HMM-VB model
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(1,1), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvbBIC(sim3[1:40], VbStructure)

# Search for the optimal number of states for HMM-VB model using
# provided values for the number of states
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(1,1), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvbBIC(sim3[1:40], VbStructure=Vb, numst=c(2L, 4L, 6L))

# Search for the optimal number of states for HMM-VB model using
# provided configurations of the number of states
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(1,1), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
configs = list(c(1,2), c(3,5), c(6,7))
```

```
hmmvbBIC(sim3[1:40], VbStructure=Vb, configList=configs)
```

HMMVBBIC-class *Class "HMMVBBIC" to represent results of HMM-VB model selection.*

Description

An S4 class to represent results of HMM-VB model selection. New instances of the class are created by hmmvbBIC.

Methods

- **show** signature(object = "HMMVBBIC") : show optimal model.
- **plot** signature(x = "HMMVBBIC", y = "missing", ...) : plot model selection results (doesn't work for configuration list provided as input to model selection).
- **getBIC** signature(object = "HMMVBBIC") : accessor for 'BIC' slot.
- **getLoglikehd** signature(object = "HMMVBBIC") : accessor for 'Loglikehd' slot.
- **getOptHMMVB** signature(object = "HMMVBBIC") : accessor for 'optHMMVB' slot.

Slots

`BIC` A numeric vector specifying calculated BIC values.

`optHMMVB` The optimal HMM-VB model with smallest BIC value.

`numst` An integer vector specifying the number of mixture components (states) in each variable block for which BIC was calculated. Number of states is the same for all variable blocks.

See Also

hmmvbBIC

---

hmmvbClust *Cluster data with HMM-VB*

Description

This function clusters dataset with HMM-VB. First, for each data point it finds an optimal state sequence using Viterbi algorithm. Next, it uses Modal Baum-Welch algorithm (MBW) to find the modes of distinct Viterbi state sequences. Data points associated the same modes form clusters. If different data sets are clustered using the same HMM-VB, clustering results of one data set can be supplied as a reference during clustering of another data set to produce aligned clusters.

Usage

```
hmmvbClust(data, model = NULL, control = clustControl(),
  rfsClust = NULL, nthread = 1, bicObj = NULL)
```

Arguments

data	A numeric vector, matrix, or data frame of observations. Categorical values are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
model	An object of class 'HMMVB' that contains trained HMM-VB obtained by the call to function hmmvbTrain.
control	A list of control parameters for clustering. The defaults are set by the call clustControl().
rfsClust	A list of parameters for the reference cluster that can be used for alignment. See HMMVBclust for details.
nthread	An integer specifying the number of threads used in clustering.
bicObj	An object of class 'HMMVBBIC' which stores results of model selection. If provided, argument model is ignored.

Value

An object of class 'HMMVBclust'.

See Also

HMMVB-class, HMMVBclust-class, hmmvbTrain

Examples

```
# cluster using trained HMM-VB
Vb <- vb(1, dim=4, numst=2)
set.seed(12345)
hmmvb <- hmmvbTrain(iris[,1:4], VbStructure=Vb)
clust <- hmmvbClust(iris[,1:4], model=hmmvb)
show(clust)
pairs(iris[,1:4], col=getClsid(clust))


# cluster using HMMVBBIC object obtained in model selection
Vb <- vb(1, dim=4, numst=1)
set.seed(12345)
modelBIC <- hmmvbBIC(iris[,1:4], VbStructure=Vb)
clust <- hmmvbClust(iris[,1:4], bicObj=modelBIC)
show(clust)
pairs(iris[,1:4], col=getClsid(clust))
```

`HMMVBclust-class` *Class "HMMVBclust" to represent clustering results with Hidden Markov Model on variable block structure.*

Description

An S4 class to represent a clustering result based on HMM-VB. New instances of the class are created by `hmmvbClust`.

Methods

- **show** signature(object = "HMMVBclust") : show clustering results based on HMM-VB.
- **plot** signature(x = "HMMVBclust", y = "missing", method = "t-sne", ...) : plot clustering results. 'method' controls the visualization algorithm. Two algorithms are supported: method = 'PCA' plots the data using 2 component PCA space; and method = 't-SNE' plots the data using 2 component t-SNE space. Default setting is t-SNE.
- **getClustParam** signature(object = "HMMVBclust") : accessor for 'clustParam' slot.
- **getLoglikehd** signature(object = "HMMVBclust") : accessor for 'Loglikehd' slot.
- **getClsid** signature(object = "HMMVBclust") : accessor for 'clsid' slot.
- **getSize** signature(object = "HMMVBclust") : accessor for 'size' slot.

Slots

`data` The input data matrix

`clustParam` A list with cluster parameters:

ncls The number of clusters (same as the number of modes)

mode A numeric matrix with cluster modes. *k*th row of the matrix stores coordinates of the *k*th mode.

ndseq The number of distinct Viterbi sequences for the dataset

vseqid An integer vector representing the map between Viterbi sequences and clusters. *k*th value in the vector stores cluster id for *k*th Viterbi sequence.

vseq A list with integer vectors representing distinct Viterbi sequences for the dataset

sigma A numeric vector with the dataset variance

`clsid` An integer vector with cluster ids.

`Loglikehd` Loglikelihood value for each data point.

`size` An integer vector with cluster sizes.

`hmmvbFindModes` *Find density modes with HMM-VB*

Description

This function finds the density modes with HMM-VB. First, for each data point it finds an optimal state sequence using Viterbi algorithm. Next, it uses Modal Baum-Welch algorithm (MBW) to find the modes of distinct Viterbi state sequences. Data points associated the same modes form clusters.

Usage

```
hmmvbFindModes(data, model = NULL, nthread = 1, bicObj = NULL)
```

Arguments

`data`	A numeric vector, matrix, or data frame of observations. Categorical values are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
`model`	An object of class 'HMMVB' that contains trained HMM-VB obtained by the call to function `hmmvbTrain`.
`nthread`	An integer specifying the number of threads used in clustering.
`bicObj`	An object of class 'HMMVBBIC' which stores results of model selection. If provided, argument `model` is ignored.

Value

An object of class 'HMMVBclust'.

See Also

`HMMVB-class`, `HMMVBclust-class`, `hmmvbTrain`

Examples

```
# find modes using trained HMM-VB
Vb <- vb(1, dim=4, numst=2)
set.seed(12345)
hmmvb <- hmmvbTrain(iris[,1:4], VbStructure=Vb)
modes <- hmmvbFindModes(iris[,1:4], model=hmmvb)
show(modes)


# find modes using HMMVBBIC object obtained in model selection
Vb <- vb(1, dim=4, numst=1)
set.seed(12345)
modelBIC <- hmmvbBIC(iris[,1:4], VbStructure=Vb)
modes <- hmmvbClust(iris[,1:4], bicObj=modelBIC)
show(modes)
```

hmmvbTrain *Train HMM-VB*

Description

This function estimates parameters for HMM-VB using the Baum-Welch algorithm. If the variable block structure is not provided, the function will first find the structure by a greedy search algorithm that minimizes BIC.

Usage

```
hmmvbTrain(data, VbStructure = NULL, searchControl = vbSearchControl(),
  trControl = trainControl(), nthread = 1)
```

Arguments

data	A numeric vector, matrix, or data frame of observations. Categorical values are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
VbStructure	An object of class 'VB'. If supplied, variable block structure stored in VbStructure is used to train HMM-VB. If not provided, a search algorithm will be perfomed to find a variable block structure with minimal BIC.
searchControl	A list of control parameters for variable block structure search. This parameter is ignored if variable block structure VbStructure is provided. The defaults are set by the call vbSearchControl().
trControl	A list of control parameters for HMM-VB training algorithm. The defaults are set by the call hmmvbTrainControl().
nthread	An integer specifying the number of threads used in searching and training routines.

Value

An object of class 'HMMVB' providing estimation for HMM-VB. The details of output components are as follows:

VbStructure	An object of class 'VB' with variable block structure for HMM-VB
HmmChain	A list of objects of class 'HMM' with trained Hidden Markov Models for each variable block.
diagCov	A logical value indicating whether or not covariance matrices for mixture models are diagonal.
BIC	BIC value for provided variable block structure or optimal BIC value for found variable block structure.

See Also

VB, vb, vbSearchControl, trainControl

Examples

```
# Train HMM-VB with known variable block structure
data("sim3")
Vb <- vb(2, dim=40, bdim=c(10,30), numst=c(3,5), varorder=list(c(1:10),c(11:40)))
set.seed(12345)
hmmvb <- hmmvbTrain(sim3[,1:40], VbStructure=Vb)
show(hmmvb)


# Train HMM-VB with unknown variable block structure using default parameters
data("sim2")
set.seed(12345)
hmmvb <- hmmvbTrain(sim2[,1:5])
show(hmmvb)

# Train HMM-VB with unknown variable block structure using with ten permutations
# and several threads
data("sim2")
set.seed(12345)
hmmvb <- hmmvbTrain(sim2[1:5], searchControl=vbSearchControl(nperm=10), nthread=3)
show(hmmvb)
```

---

sim2 *Synthetic dataset used in section 5.1.2 of the reference paper.*

Description

Dataset used for testing clustering with HMM-VB. The data dimension is 5. Data points were drawn from a 10-component Gaussian Mixture Model. By specific choice of the means, the data contains 10 distinct clusters. For details see the references.

Usage

```
sim2
```

Format

A data frame with 5000 rows and 5 variables. Last column contains ground truth cluster labels.

References

Lin Lin and Jia Li, "Clustering with hidden Markov model on variable blocks," **Journal of Machine Learning Research**, 18(110):1-49, 2017.

sim3 *Synthetic dataset used in section 5.1.3 of the reference paper*

Description

Dataset used for testing clustering with HMM-VB. The data dimension is 40. The first 10 dimensions were generated from a 3-component Gaussian Mixture Model (GMM). The remaining 30 dimensions were generated from a 5-component GMM. By specific design of the means, covariance matrices and transition probabilities, the data contain 5 distinct clusters. For details see the references.

Usage

```
sim3
```

Format

A data frame with 1000 rows and 40 variables. Last column contains ground truth cluster labels.

References

Lin Lin and Jia Li, "Clustering with hidden Markov model on variable blocks," **Journal of Machine Learning Research**, 18(110):1-49, 2017.

trainControl *Parameters for HMM-VB training.*

Description

This function creates a list with parameters for estimating an HMM-VB, which is used as an argument for `hmmvbTrain`.

Usage

```
trainControl(ninit0 = 1, ninit1 = 0, ninit2 = 0, epsilon = 1e-04,
  diagCov = FALSE)
```

Arguments

`ninit0`	The number of initializations for default scheme 0, under which the k-means clustering for entire dataset is used to initialize the model.
`ninit1`	The number of initializations for default scheme 1, under which the k-means clustering for a subset of data is used to initialize the model.
`ninit2`	The number of initializations for default scheme 2, under which a random subset of data is used as cluster centroids to initialize the model.

epsilon	Stopping criteria for Baum-Welch algorithm. Should be a small number in range (0,1).
diagCov	A logical value indicating whether or not variable block covariance matrices will be diagonal.

Value

The named list with parameters.

See Also

hmmvbTrain

Examples

```
# setting up multiple initialization schemes
Vb <- vb(1, dim=4, numst=2)
set.seed(12345)
hmmvb <- hmmvbTrain(iris[,1:4], VbStructure=Vb,
          trControl=trainControl(ninit0=2, ninit1=2, ninit2=2))
show(hmmvb)

# forcing diagonal covariance matrices
Vb <- vb(1, dim=4, numst=2)
set.seed(12345)
hmmvb <- hmmvbTrain(iris[,1:4], VbStructure=Vb,
          trControl=trainControl(diagCov=TRUE))
show(hmmvb)
```

---

vb *Make an instance of class "VB"*

Description

This function creates a variable block structure.

Usage

```
vb(nb, dim, bdim = NULL, numst, varorder = NULL)
```

Arguments

nb	The number of variable blocks.
dim	Dimensionality of the data.
bdim	An integer vector specifying dimensionality of each variable block. This argument can be omitted if the variable block structure has a single block (case of GMM).

`numst` An integer vector specifying the number of mixture models in each variable block.

`varorder` A list of integer vectors specifying the variable order in each variable block. This argument can be omitted if variable structure has a single variable block (GMM).

Value

An object of class "VB".

See Also

VB

Examples

```
# variable block structure for GMM with 3 dimensions and 2 mixture states
Vb <- vb(1, dim=3, numst=2)

# variable block structure with 2 variable blocks
Vb <- vb(2, dim=10, bdim=c(4,6), numst=c(3,11), varorder=list(c(1:4),c(5:10)))
```

VB-class *Class "VB" to represent a variable block structure.*

Description

An S4 class to represent a variable block structure. To create a new instance of the class, use vb.

Methods

- **show** signature(object = "VB") : show parameters of variable blocks structure.
- **getNb** signature(object = "VB") : accessor for 'nb' slot.
- **getDim** signature(object = "VB") : accessor for 'dim' slot.
- **getBdim** signature(object = "VB") : accessor for 'bdim' slot.
- **getNumst** signature(object = "VB") : accessor for 'numst' slot.
- **getVarorder** signature(object = "VB") : accessor for 'varorder' slot.

Slots

`nb` The number of variable blocks.

`dim` Dimensionality of the data.

`bdim` An integer vector specifying dimensionality of each variable block.

`numst` An integer vector specifying the number of mixture models in each variable block.

`varorder` A list of integer vectors specifying the variable order in each variable block.

vbSearchControl *Parameters for variable block structure search.*

Description

This function creates a list with parameters for the search of a variable block structure used as an argument for hmmvbTrain.

Usage

```
vbSearchControl(perm = NULL, numstPerDim = NULL, dim = NULL,
  maxDim = 10, minDim = 1, nperm = 1, relax = FALSE)
```

Arguments

perm	A list of integer vectors specifying variable permutations. If provided, the argument dim must be supplied.
numstPerDim	An integer vector of length maxDim specifying a map from the variable block dimensionality to the number of states in the block. *k*th value in the vector corresponds to number of states for dimensionality *k*.
dim	Data dimensionality. Must be provided with perm argument, otherwise is ignored.
maxDim	Maximum variable block dimension.
minDim	Minimum variable block dimension. Should be an integer equal to 1 or 2.
nperm	The number of variable permutations. This parameter is ignored if permutations are provided in perm argument.
relax	A logical value indicating whether or not variable block structure search will be performed under less restricting conditions.

Value

The named list with parameters.

See Also

hmmvbTrain

Examples

```
# setting up permutations
perm <- list(c(1,2,3), c(1,3,2), c(3,2,1))
searchControl <- vbSearchControl(perm=perm, dim=3)

# setting up a map between block dimensionality and number of states
searchControl <- vbSearchControl(maxDim=5, numstPerDim=c(3,4,5,6,7))
```

Index