

# Package ‘GBClust’

November 27, 2020

**Type** Package

**Title** Generalized Bayes clustering

**Version** 0.0.2

**Date** 2020-11-27

**Description** This package is an implementation of several generalized Bayes clustering methods.

**Encoding** UTF-8

**License** MIT + file LICENSE

**LazyData** TRUE

**Imports** Rcpp (>= 1.0.2), ggplot2, cluster

**LinkingTo** Rcpp, RcppArmadillo

**RoxygenNote** 7.1.1

**Suggests** knitr,  
rmarkdown

**VignetteBuilder** knitr

## R topics documented:

comp_medoids . . . . .	2
kbinary . . . . .	2
kbinary_gibbs . . . . .	3
kbinary_select . . . . .	3
kdiss . . . . .	4
kdiss_select . . . . .	5
kmeans2 . . . . .	5
kmeans2_select . . . . .	6
kmeans_gibbs . . . . .	6
Minkowski_gibbs . . . . .	7
<b>Index</b>	<b>9</b>

---

comp_medoids	<i>Computation of the medoids</i>
--------------	-----------------------------------

---

**Description**

Compute the medoids of a given clustering solution based on the corresponding dissimilarity matrix.

**Usage**

```
comp_medoids(D, cluster)
```

**Arguments**

D	A n x n numeric matrix containing the dissimilarities, i.e. the output of the functions <a href="#">dist</a> or <a href="#">daisy</a> .
cluster	A clustering solution, i.e. the output of <a href="#">kdiss</a> .

**Value**

medoids Indexes of the medoids.

---

kbinary	<i>K-binary clustering</i>
---------	----------------------------

---

**Description**

Perform the so-called k-binary clustering algorithm, for obtaining groups when the data are binary observations.

**Usage**

```
kbinary(x, k, nstart = 1, trace = FALSE)
```

**Arguments**

x	binary matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns).
k	The number of clusters to be considered. A random set of (distinct) rows in x is chosen as the initial centres.
nstart	Number of random sets that has been chosen
trace	logical: if true, tracing information on the progress of the algorithm is produced.

**Value**

cluster	Labels of the clusters at convergence
centers	The value of the centroids at convergence
loss	Numeric value of the loss function at convergence

---

kbinary_gibbs	<i>K-binary Gibbs sampling</i>
---------------	--------------------------------

---

**Description**

Perform the Gibbs-sampling for the k-binary algorithm. This function is complementary to [kbinary](#), which is used instead to get a point estimate.

**Usage**

```
kbinary_gibbs(
  x,
  k,
  lambda = 1,
  R = 1000,
  burn_in = 1000,
  nstart = 10,
  trace = FALSE
)
```

**Arguments**

x	binary matrix of the data
k	The number of clusters to be considered
lambda	Gibbs posterior tuning parameter
R	Number of MCMC samples after burn-in
burn_in	Number of MCMC samples to be discarded as burn-in period
nstart	Number of random initializations for the k-means algorithm
trace	logical: if true, tracing information on the progress of the algorithm is produced.

**Value**

G A R x n matrix including the cluster labels for each MCMC iteration.  
 loss A R-dimensional vector including the values of the loss function for each MCMC iteration.  
 G\_map Labels of the clusters at the lowest value of the posterior that has been computed.  
 loss\_map Lowest value of the loss that has been computed.

---

kbinary_select	<i>Selection of the number of cluster for the k-binary algorithm</i>
----------------	--

---

**Description**

It displays the value of the loss function for various choices of k.

**Usage**

```
kbinary_select(x, k_max, nstart = 1)
```

**Arguments**

<code>x</code>	numeric matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns).
<code>k_max</code>	The maximum number of clusters to be considered. A random set of (distinct) rows in <code>x</code> is chosen as the initial centres.
<code>nstart</code>	Number of random sets that has been chosen

**Value**

It plots the loss function for different clustering solutions

---

<code>kdiss</code>	<i>K-dissimilarities algorithm</i>
--------------------	------------------------------------

---

**Description**

Perform the so-called k-dissimilarities algorithm.

**Usage**

```
kdiss(D, k, nstart = 1, trace = FALSE)
```

**Arguments**

<code>D</code>	A $n \times n$ numeric matrix containing the dissimilarities, typically the output of <a href="#">dist</a> or <a href="#">daisy</a> .
<code>k</code>	The number of clusters to be considered. See <a href="#">kdiss_select</a> for selection criteria.
<code>nstart</code>	Number of random initializations.
<code>trace</code>	logical: if true, tracing information on the progress of the algorithm is produced.

**Value**

`cluster` Labels of the clusters at convergence

`loss` Numeric value of the loss function at convergence

---

kdiss_select	<i>Selection of the number of cluster for the k-dissimilarities algorithm</i>
--------------	---

---

**Description**

It displays the value of the loss function / average silhouette width, for different values of k

**Usage**

```
kdiss_select(D, k_max, nstart = 1, method = "elbow")
```

**Arguments**

D	A $n \times n$ numeric matrix containing the dissimilarities, typically the output of <a href="#">dist</a> or <a href="#">daisy</a> .
k_max	Maximum number of clusters to be considered.
nstart	Number of random initializations.
method	The graph that will be displayed. Supported options are method="elbow", which displays the loss function, or method="silhouette". See <a href="#">silhouette</a> for details about the latter.

**Value**

It return a [ggplot2](#) graph of the loss function / average silhouette width, for  $k=1, \dots, k_{\max}$ .

---

kmeans2	<i>K-Means clustering</i>
---------	---------------------------

---

**Description**

Perform the k-means clustering on a data matrix.

**Usage**

```
kmeans2(x, k, nstart = 1, algorithm = "kmeans", trace = FALSE)
```

**Arguments**

x	numeric matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns).
k	The number of clusters to be considered. A random set of (distinct) rows in x is chosen as the initial centres.
nstart	Number of random sets that has been chosen
algorithm	The algorithm to be used
trace	logical: if true, tracing information on the progress of the algorithm is produced.

**Value**

cluster Labels of the clusters at convergence  
 centers The value of the centroids at convergence  
 loss Numeric value of the loss function at convergence

---

kmeans2_select	<i>Selection of the number of cluster for the k-dissimilarities algorithm</i>
----------------	---

---

**Description**

It displays the value of the loss function for various choices of k

**Usage**

```
kmeans2_select(x, k_max, nstart = 1, algorithm = "kmeans")
```

**Arguments**

x numeric matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns).  
 k\_max The maximum number of clusters to be considered. A random set of (distinct) rows in x is chosen as the initial centres.  
 nstart Number of random sets that has been chosen  
 algorithm The algorithm to be used, either kmeans or kmeans2

**Value**

It plots the loss function for different clustering solutions

---

kmeans_gibbs	<i>K-means Gibbs sampling</i>
--------------	-------------------------------

---

**Description**

Perform the Gibbs-sampling for the k-means algorithm. This function is complementary to [kmeans2](#), which is used instead to get a point estimate.

**Usage**

```
kmeans_gibbs(
  x,
  k,
  a_lambda,
  b_lambda,
  R = 1000,
  burn_in = 1000,
  nstart = 10,
  trace = FALSE
)
```

**Arguments**

x	A n x d numeric matrix of the data.
k	The number of clusters to be considered.
a_lambda	Hyperparameter of the Gamma prior on the scale parameter.
b_lambda	Hyperparameter of the Gamma prior on on the scale parameter.
R	Number of MCMC samples after burn-in.
burn_in	Number of MCMC samples to be discarded as burn-in period.
nstart	Number of random initializations for the k-means algorithm.
trace	logical: if true, tracing information on the progress of the algorithm is produced.

**Value**

G A R x n matrix including the cluster labels for each MCMC iteration.  
lambda A R-dimensional vector including the values of lambda for each MCMC iteration.  
loss A R-dimensional vector including the values of the loss function for each MCMC iteration.  
G\_map Labels of the clusters at the lowest value of the posterior that has been computed.  
loss\_map Lowest value of the loss that has been computed.

---

Minkowski_gibbs	<i>K-dissimilarities (Minkowski) Gibbs sampling</i>
-----------------	---

---

**Description**

Perform the Gibbs-sampling for the k-dissimilarities algorithm using the Minkowski distance. This function is complementary to [kdiss](#), which is used instead to get a point estimate.

**Usage**

```
Minkowski_gibbs(
  x,
  k,
  p,
  a_lambda = 0,
  b_lambda = 0,
  R = 1000,
  burn_in = 1000,
  nstart = 10,
  trace = FALSE
)
```

**Arguments**

x	numeric matrix of of the data.
k	The number of clusters to be considered.
p	Power of the Minkowski distance.
a_lambda	Hyperparameter of the Gamma prior on the scale parameter. The default a_lambda = 0 leads to an improper prior.

b_lambda	Hyperparameter of the Gamma prior on the scale parameter. The default $a_{\text{lambda}} = 0$ leads to an improper prior.
R	Number of MCMC samples after burn-in.
burn_in	Number of MCMC samples to be discarded as burn-in period.
nstart	Number of random initializations for the <a href="#">kdiss</a> algorithm, used to initialize the MCMC chain.
trace	logical: if true, tracing information on the progress of the algorithm is produced.

**Value**

G Labels of the clusters at each MCMC iteration.

lambda Numeric vector of the values of lambda at each MCMC iteration.

loss Numeric vector of the loss function at each MCMC iteration.

G\_map Labels of the clusters obtained using [kdiss](#), representing the maximum a posteriori.

loss\_map Numeric value of the loss function obtained using [kdiss](#), representing the maximized loss.



# Index

comp\_medoids, [2](#)

daisy, [2](#), [4](#), [5](#)

dist, [2](#), [4](#), [5](#)

ggplot2, [5](#)

kbinary, [2](#), [3](#)

kbinary\_gibbs, [3](#)

kbinary\_select, [3](#)

kdiss, [2](#), [4](#), [7](#), [8](#)

kdiss\_select, [4](#), [5](#)

kmeans2, [5](#), [6](#)

kmeans2\_select, [6](#)

kmeans\_gibbs, [6](#)

Minkowski\_gibbs, [7](#)

silhouette, [5](#)