

# Package: bigMap (via r-universe)

November 7, 2024

**Type** Package

**Title** Big Data Mapping

**Version** 4.6.2

**Date** 2024-06-01

**Description** Unsupervised clustering protocol for large scale structured data, based on a low dimensional representation of the data. Dimensionality reduction is performed using a parallelized implementation of the t-Stochastic Neighboring Embedding algorithm (Garriga J. and Bartumeus F. (2018), [arXiv:1812.09869](https://arxiv.org/abs/1812.09869)).

**License** GPL-3 | file LICENSE

**Depends** R (>= 3.5.0)

**Imports** Rcpp (>= 0.12.0), bigmemory (>= 4.5.0), parallel (>= 3.5.0), RColorBrewer, colorspace,

**Suggests** snow (>= 0.4-2), knitr, rmarkdown

**LinkingTo** Rcpp, RcppArmadillo, BH, bigmemory

**LazyData** FALSE

**VignetteBuilder** knitr

**RoxygenNote** 7.3.1

**SystemRequirements** GNU make

**NeedsCompilation** yes

**Author** Joan Garriga [aut, cre], Frederic Bartumeus [aut]

**Maintainer** Joan Garriga <jgarriga@ceab.csic.es>

**Config/pak/sysreqs** make

**Repository** <https://jgarriga65.r-universe.dev>

**RemoteUrl** <https://github.com/jgarriga65/bigmap>

**RemoteRef** HEAD

**RemoteSha** bec112db4a3619893cd16a33952f209170cb3cb1

## Contents

bdm.boxp . . . . .	2
bdm.cost . . . . .	3
bdm.dMap . . . . .	4
bdm.dMap.plot . . . . .	5
bdm.example . . . . .	7
bdm.hlCorr . . . . .	7
bdm.init . . . . .	8
bdm.knp . . . . .	9
bdm.knp.plot . . . . .	10
bdm.labels . . . . .	11
bdm.merge.s2nr . . . . .	12
bdm.mpi.start . . . . .	13
bdm.mpi.stop . . . . .	13
bdm.mtsne . . . . .	14
bdm.optk.plot . . . . .	15
bdm.optk.s2nr . . . . .	16
bdm.pakde . . . . .	17
bdm.pakde.plot . . . . .	18
bdm.pMap . . . . .	19
bdm.ptsne . . . . .	20
bdm.ptsne.plot . . . . .	21
bdm.qMap . . . . .	22
bdm.restart . . . . .	23
bdm.wtt . . . . .	24
bdm.wtt.plot . . . . .	25
<b>Index</b>	<b>27</b>

---

bdm.boxp	<i>Clustering statistics box-plot.</i>
----------	--

---

### Description

Clustering statistics box-plot.

### Usage

```
bdm.boxp(data, bdm, byVars = F, merged = T, clusters = NULL, layer = 1)
```

### Arguments

data	A matrix of data to be plotted (either the input data matrix or any covariate matrix as well).
bdm	A <i>bdm</i> instance as generated by <code>bdm.init()</code> .
byVars	A logical value. By default ( <code>byVars = FALSE</code> ) box-plots are grouped by cluster. With <code>byVars = TRUE</code> box-plots are grouped by input feature.

merged	A logical value. If TRUE (default value) and the <code>!is.null(bdm\$merge)</code> the boxplots depict the clusters after merging. If <code>merged = FALSE</code> or <code>is.null(bdm\$merge)</code> the boxplots correspond to the top-level clustering.
clusters	A vector with a subset of cluster ids. (Default value is <code>clusters=NULL</code> to plot all clusters, with a maximum of 25).
layer	The number of a layer (1 by default).

### Details

If the number of clusters is large, only the first 25 clusters will be plotted. Note that the WTT algorithm numbers the clusters based on density value at the peak cell of the cluster. Thus, the numbering of the clusters is highly correlated with their relevance in terms of partial density. Therefore, in case of more than 25 clusters, the most relevant should always be included in the plot.

### Value

None.

### Examples

```
bdm.example()
bdm.bxp(ex$map, data = ex$data[, 1:4])
bdm.bxp(ex$map, data = ex$data[, 1:4], byVars = TRUE)
```

---

bdm.cost	<i>ptSNE cost &amp; size plot.</i>
----------	------------------------------------

---

### Description

ptSNE cost & size plot.

### Usage

```
bdm.cost(bdm, x.lim = NULL)
```

### Arguments

bdm	A <i>bdm</i> instance as generated by <code>bdm.init()</code> or a list of them to make a comparative plot.
x.lim	A vector with upper and lower bounds to limit the number of epochs in the x-axis (NULL by default).

### Value

None.

## Examples

```
bdm.example()
bdm.cost(ex$map)
```

---

bdm.dMap	<i>Class density maps</i>
----------	---------------------------

---

## Description

Compute the class density maps of a set of classes on the embedding grid. This function returns a fuzzy mapping of the set of classes on the grid cells. The classes can be whatever set of classes of interest and must be given as a vector of point-wise discrete labels (either numeric, string or factor).

## Usage

```
bdm.dMap(bdm, data = NULL, threads = 2, mpi.cl = NULL, layer = 1)
```

## Arguments

bdm	A <i>bdm</i> instance as generated by <code>bdm.init()</code> .
data	A vector of discret covariates or class labels. The covariate values can be of any factorizable type. By default ( <code>data=NULL</code> ) the function computes the density maps based on the clustering labels (i.e. equivalent to <code>data=bdm.labels(bdm)</code> ).
threads	Number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory. Default value is <code>threads = 4</code> ).
mpi.cl	MPI (inter-node parallelization) cluster as generated by <code>bdm.mpi.start()</code> . (By default <code>mpi.cl = NULL</code> a 'SOCK' (intra-node parallelization) cluster is generated).
layer	The number of the t-SNE layer (1 by default).

## Details

`bdm.dMap()` computes the join distribution  $P(V = v_i, C = c_j)$  where  $V = v_1, \dots, v_l$  is the discrete covariate and  $C = c_1, \dots, c_g$  are the grid cells of the paKDE raster. That is, this function recomputes the paKDE but keeping track of the covariate (or class) label of each data-point. This results in a fuzzy distribution of the covariate (class) at each cell.

Usually, figuring out the join distribution  $P(V = v_i, C = c_j)$  entails an intensive computation. Thus `bdm.dMap()` performs the computation and stores the result in a dedicated element named `$dMap`. Afterwards the class density maps can be visualized with the `bdm.dMap.plot()` function.

## Value

A copy of the input *bdm* instance with element `$dMap`, a matrix with a soft clustering of the grid cells.

**Examples**

```
# --- load example dataset
bdm.example()
## Not run:
m <- bdm.dMap(ex$map, threads = 4)

## End(Not run)
```

---

bdm.dMap.plot	<i>Class density maps plot.</i>
---------------	---------------------------------

---

**Description**

Class density maps plot.

**Usage**

```
bdm.dMap.plot(
  bdm,
  classes = NULL,
  join = FALSE,
  class.pltt = NULL,
  pakde.pltt = NULL,
  pakde.lvls = 16,
  wtt.lwd = 1,
  plot.peaks = T,
  labels.cex = 1,
  layer = 1
)
```

**Arguments**

<b>bdm</b>	A <i>bdm</i> instance as generated by <code>bdm.init()</code> .
<b>classes</b>	A vector with a subset of class names or covariate values. Default value is <code>classes=NULL</code> . If no classes are specified (default value) all classes are plotted.
<b>join</b>	Logical value. If <code>FALSE</code> (default value), class mapping is based on the class conditional distributions. If <code>TRUE</code> , class mapping is based on the overall classes join distribution.
<b>class.pltt</b>	A colour palette to show class labels in the hard mapping. By default ( <code>class.pltt = NULL</code> ) the default palette is used.
<b>pakde.pltt</b>	A palette of colours to indicate the levels of the class density maps. The length of the colour palette should be at least the number of levels specified in <i>pakde.lvls</i> .
<b>pakde.lvls</b>	The number of levels of the heat-map when plotting class density maps (16 by default).
<b>wtt.lwd</b>	The width of the watertrack lines (as set in <code>par()</code> ).

<code>plot.peaks</code>	Logical value (TRUE by default). If set to TRUE and the up-stream step <code>bdm\$wt.t()</code> is computed the peak of each cluster is depicted.
<code>labels.cex</code>	If <code>plot.peaks</code> is TRUE, the size of the labels of the clusters (as set in <code>par()</code> ). By default <code>labels.cex=0.0</code> and the labels of the clusters are not depicted.
<code>layer</code>	The number of the layer from which the class density maps are computed (1 by default).

## Details

`bdm.dMap.plot()` yields a multi-plot layout where the first plot shows the dominating value of the covariate (or dominating class) in each cell, and the rest of the plots show the density map of each covariate value (or class).

The join distribution  $P(V = v_i, C = c_j)$  is prone to the bias in the marginal distribution of the covariate. Therefore, the join distribution  $P(V = v_i, C = c_j)$  is transformed, by default, into a conditional distribution  $P(c_j|V = v_i)$  (where the  $c_j$  are the grid cells of the embedding and  $V$  is the covariate (or class)). Thus, the first plot shows a hard classification of grid-cells, (cells are coloured based on the dominating value of the covariate (or dominating class), *i.e.* the  $v_i$  for which  $P(c_j|V = v_i)$  is maximum), and the rest of the plots show the conditional distributions  $P(C = c_j|V = v_i)$ . This makes the plots of the different classes not directly comparable but the dominant areas of each class can be more easily identified.

However, the same plots can be depicted based on the join distribution by setting `join = TRUE`. This makes sense when the bias in the covariate values (or classes) is not significant. In this case the hard clustering shows the real dominance of each covariate value (or class) over the embedding area and the density maps are comparable one to each other (although, individually, they are not real density functions as they do not add up to one).

The multi-plot layout can be limited to a subset of the values of the covariate (or subset of classes) specified in parameter `classes`.

## Value

None.

## Examples

```
# --- load example dataset
bdm.example()
## Not run:
m <- bdm.dMap(ex$map, threads = 4)
bdm.dMap.plot(m)

## End(Not run)
```

bdm.example

*Example dataset***Description**

Loads a mapping example.

**Usage**

```
bdm.example()
```

**Details**

The object `ex` is a list with elements: `ex$data`, a matrix with raw data; `ex$labels`, a vector of datapoint labels; `ex$map`, a *bdm* data mapping instance. A *bdm* instance is the basic object of the mapping protocol, i.e. a list to which new elements are added at each step of the mapping protocol.

This example is based on a small synthetic dataset with  $n = 5000$  observations drawn from a 4-variate Gaussian Mixture Model (GMM) with 16 Gaussian components with random means and variances.

**Value**

An example dataset named `ex`

**Examples**

```
# --- load example dataset
bdm.example()
str(ex)
```

bdm.hlCorr

*HD/LD correlation.***Description**

Pair-wise distance correlation between HD and LD neighborhoods.

**Usage**

```
bdm.hlCorr(data, bdm, zSampleSize = 1000, threads = 4, mpi.cl = NULL)
```

**Arguments**

<code>data</code>	Input data (a matrix, a <code>big.matrix</code> or a <code>.csv</code> file name).
<code>bdm</code>	A <i>bdm</i> instance as generated by <code>bdm.ptsne()</code> .
<code>zSampleSize</code>	Number of data points to check by thread. (Default value is <code>zSampleSize=1000</code> ).
<code>threads</code>	The number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory. Default value is <code>threads = 4</code> ).
<code>mpi.cl</code>	An MPI (inter-node parallelization) cluster as generated by <code>bdm.mpi.start()</code> . (By default <code>mpi.cl = NULL</code> a 'SOCK' (intra-node parallelization) cluster is generated).

**Value**

A copy of the input *bdm* instance with new element *bdm\$knP*.

**Examples**

```
# --- load example dataset
## Not run:
bdm.example()
m <- bdm.hlCorr(exData[, 1:4], ex$map, threads = 4)

## End(Not run)
```

---

bdm.init

*Embedding initialization.*


---

**Description**

Computes the precision parameters for the given perplexity (i.e. the local bandwidths for the input affinity kernels) and returns them as a *bdm* data mapping instance. A *bdm* data mapping instance is the starting object of the mapping protocol, a list to which new elements are added at each step of the mapping protocol.

**Usage**

```
bdm.init(
  data,
  is.distance = F,
  is.sparse = F,
  ppx = 100,
  mpi.cl = NULL,
  threads = 4,
  labels = NULL
)
```



**Arguments**

<code>data</code>	A <i>data.frame</i> or <i>matrix</i> with raw input-data. The dataset must not have duplicated rows.
<code>is.distance</code>	Default value is <i>is.distance</i> = <i>FALSE</i> . TRUE indicates that raw data is a distance matrix.
<code>is.sparse</code>	Default value is <i>is.sparse</i> = <i>FALSE</i> . TRUE indicates that the raw data is a sparse matrix.
<code>ppx</code>	The value of perplexity to compute similarities.
<code>mpi.cl</code>	An MPI (inter-node parallelization) cluster as returned by <i>bdm.mpi.start()</i> . By default <i>mpi.cl</i> = <i>NULL</i> , i.e. a 'SOCK' (intra-node parallelization) cluster is used.
<code>threads</code>	Number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory. Default value is <i>threads</i> = 4).
<code>labels</code>	If available, a length <i>nrow(data)</i> vector of class labels. Label values are factorized as <i>as.numeric(as.factor(labels))</i> .

**Value**

A *bdm* data mapping instance. This *bdm* instance is the starting object of the mapping protocol: a list to which new elements are added at each step of the mapping protocol.

**Examples**

```
# --- load example dataset
bdm.example()
## Not run:
m <- bdm.init(ex$data, ppx = 250, labels = ex$labels)

## End(Not run)
```

bdm.knp

*k*-ary Neighborhood Preservation**Description**

A measure of matching between HD and LD neighborhoods ('Multi-scale similarities in stochastic neighbour embedding: Reducing dimensionality while preserving both local and global structure', Lee et. al 2015).

**Usage**

```
bdm.knp(data, bdm, k.max = NULL, sampling = 0.9, threads = 4, mpi.cl = NULL)
```

**Arguments**

data	Input data (a matrix, a big.matrix or a .csv file name).
bdm	A <i>bdm</i> instance as generated by <code>bdm.ptsne()</code> .
k.max	Maximum neighborhood size to check. (By default <code>k.max=NULL</code> neighborhood sizes are checked up to <code>n-1</code> ).
sampling	Fraction of data points to check for each neighborhood size. (Default value is <code>sampling=0.9</code> ).
threads	The number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory. Default value is <code>threads = 4</code> ).
mpi.cl	An MPI (inter-node parallelization) cluster as generated by <code>bdm.mpi.start()</code> . (By default <code>mpi.cl = NULL</code> a 'SOCK' (intra-node parallelization) cluster is generated).

**Value**

A copy of the input *bdm* instance with new element *bdm\$knP*. #'

**Examples**

```
# --- load example dataset
bdm.example()
## Not run:
# --- compute the knP
m <- bdm.knp(ex$data, ex$map, threads = 4)
# --- plot the knP
bdm.knp.plot(m)

## End(Not run)
```

---

bdm.knp.plot

*k-ary Neighborhood Preservation plot*


---

**Description**

Log and linear plots of the *k*-ary Neighborhood Preservation

**Usage**

```
bdm.knp.plot(bdm, ppxfmt = 0)
```

**Arguments**

bdm	A <i>bdm</i> data mapping instance, or a list of them to make a comparative plot.
ppxfmt	Format of <i>ppx</i> in the legend. If <code>ppxfmt &gt; 1</code> then <i>ppx</i> is shown as a fraction of the data set size and <i>ppxfmt</i> is the number of decimal digits. Default value is <code>ppxfmt = 0</code> and then <i>ppx</i> is expressed in absolute value.

**Examples**

```
# --- load example dataset
bdm.example()
## Not run:
# --- compute the kNP
m <- bdmm.knp(ex$data, ex$map, threads = 4)
# --- plot the kNP
bdm.knp.plot(m, ppxfmt = 0)

## End(Not run)
```

---

bdm.labels	<i>Get data-point clustering labels.</i>
------------	--

---

**Description**

Given that clusters are computed at grid-cell level, this function returns the clustering label for each data-point.

**Usage**

```
bdm.labels(bdm, merged = F, layer = 1)
```

**Arguments**

bdm	A <i>bdm</i> data mapping instance.
merged	Default value is <i>merged = FALSE</i> . If <i>merged = TRUE</i> and the clustering has been merged, the labels are the ids of the clusters after merging. If <i>merged = FALSE</i> or the clustering has not been merged, the labels indicate the ids of to the top-level clustering.
layer	The ptSNE output layer. Default value is <i>layer = 1</i> .

**Value**

A vector of data-point clustering labels.

**Examples**

```
bdm.example()
m.labels <- bdmm.labels(ex$map)
```

---

bdm.merge.s2nr

Merging of clusters based on signal-to-noise-ratio.

---

## Description

Performs a recursive merging of clusters based on minimum loss of signal-to-noise-ratio (S2NR) until reaching the desired number of clusters. The S2NR is the explained/unexplained variance ratio measured in the high dimensional space based on the given low dimensional clustering.

## Usage

```
bdm.merge.s2nr(
  data,
  bdm,
  k = 10,
  plot.merge = T,
  ret.merge = F,
  info = T,
  layer = 1,
  ...
)
```

## Arguments

data	Input data (a matrix, a big.matrix or a .csv file name).
bdm	A <i>bdm</i> instance as generated by <code>bdm.init()</code> .
k	The number of desired clusters. The clustering will be recursively merged until reaching this number of clusters (default value is <code>k = 10</code> ). By setting <code>k &lt; 0</code> we can specify the number of clusters that we are willing to merge.
plot.merge	Logical value. If TRUE, the merged clustering is plotted (default value is <code>plot.merge = TRUE</code> )
ret.merge	Logical value. If TRUE, the function returns a copy of the input <i>bdm</i> instance with the merged clustering attached as <i>bdm\$merge</i> (default value is <code>ret.merge = FALSE</code> )
info	Logical value. If TRUE, all merging steps are shown (default value is <code>info = FALSE</code> ).
layer	The <i>bdm\$ptsne</i> layer to be used (default value is <code>layer = 1</code> ).
...	If <i>plot.merge</i> is TRUE, you can set the <code>bdm.wtt.plot()</code> parameters to control the plot.

## Details

See details in `bdm.optk.s2nr()`.

**Value**

None if `ret.merge = FALSE`. Else, a copy of the input *bdm* instance with new element *bdm\$merge*.

**Examples**

```
bdm.example()
m.labels <- bdm.labels(ex$map)
```

---

<code>bdm.mpi.start</code>	<i>Initialize parallel computing environment.</i>
----------------------------	---

---

**Description**

Initialize parallel computing environment.

**Usage**

```
bdm.mpi.start(threads)
```

**Arguments**

<code>threads</code>	The number of parallel threads (in principle only limited by hardware resources, i.e. number of cores and available memory)
----------------------	---

**Value**

`cl` A cluster instance (as created by the `snow::makeCluster()` function).

---

<code>bdm.mpi.stop</code>	<i>Stops MPI parallel computing environment.</i>
---------------------------	--

---

**Description**

Stops MPI parallel computing environment.

**Usage**

```
bdm.mpi.stop(cl)
```

**Arguments**

<code>cl</code>	A cluster instance (as created by the <code>bdm.mpi.start()</code> function).
-----------------	---

---

bdm.mtsne	<i>Multi-core t-SNE (mtSNE)</i>
-----------	---------------------------------

---

### Description

Starts the multi-core t-SNE (mtSNE) algorithm.

### Usage

```
bdm.mtsne(
  data,
  is.distance = F,
  is.sparse = F,
  ppx = 100,
  theta = 0.5,
  iters = 250,
  mpi.cl = NULL,
  threads = 4,
  infoRate = 25
)
```

### Arguments

data	A <i>data.frame</i> or <i>matrix</i> with raw input-data. The dataset must not have duplicated rows.
is.distance	Default value is <i>is.distance = FALSE</i> . TRUE indicates that raw data is a distance matrix.
is.sparse	Default value is <i>is.sparse = FALSE</i> . TRUE indicates that the raw data is a sparse matrix.
ppx	The value of perplexity to compute similarities.
theta	Accuracy/speed trade-off factor, a value between 0.33 and 0.8. Default value is <i>theta = 0.5</i> . If <i>theta &lt; 0.33</i> the algorithm uses the exact computation of the gradient. The closer it is this value to 1 the faster the computation and the coarser the approximation of the gradient.
iters	Number of iters/epoch. Default value is <i>iters = 250</i> .
mpi.cl	An MPI (inter-node parallelization) cluster as generated by <i>bdm.mpi.start()</i> . By default <i>mpi.cl = NULL</i> , i.e. a 'SOCK' (intra-node parallelization) cluster is generated.
threads	Number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory). Default value is <i>threads = 4</i> .
infoRate	Number of epochs to show output information. Default value is <i>infoRate = 25</i> .

### Value

A *bdm* data mapping instance.

**Examples**

```
# --- load example dataset
bdm.example()
## Not run:
# --- perform mtSNE
m <- bdmsne(ex$data, ex$map, ppx = 250, iters = 250, threads = 4)
# --- plot the Cost function
bdm.cost(m)
# --- plot mtSNE output (use bdmsne.plot() function)
bdmsne.plot(m)

## End(Not run)
```

---

bdm.optk.plot

*Plots the signal-to-noise-ratio as a function of the number of clusters.*


---

**Description**

The function `bdm.optk.sn2r()` computes the S2NR that results from recursively merging clusters and, by default, makes a plot of these values. For large datasets this computation can take a while, so we can save this result by setting `optk.ret = TRUE`. If this result is saved, we can plot it again at any time using this function.

**Usage**

```
bdm.optk.plot(bdm)
```

**Arguments**

`bdm`                      A *bdm* instance as generated by `bdm.init()`.

**Value**

None.

**Examples**

```
bdm.example()
m <- bdmsne(ex$data, ex$map, ret.optk = TRUE)
bdmsne.plot(m)
```

---

bdm.optk.s2nr

Find optimal number of clusters based on signal-to-noise-ratio.

---

## Description

Performs a recursive merging of clusters based on minimum loss of signal-to-noise-ratio (S2NR). The S2NR is the explained/unexplained variance ratio measured in the high dimensional space based on the given low dimensional clustering. Merging is applied recursively until reaching a configuration of only 2 clusters and the S2NR is measured at each step.

## Usage

```
bdm.optk.s2nr(data, bdm, info = T, plot.optk = T, ret.optk = F, layer = 1)
```

## Arguments

data	Input data (a matrix, a big.matrix or a .csv file name).
bdm	A clustered <i>bdm</i> instance ( <i>i.e.</i> all up-stream steps performed: <code>bdm.ptse()</code> , <code>bdm.pakde()</code> and <code>bdm.wtt()</code> ).
info	Logical value. If TRUE, all merging steps are shown (default value is <code>info = FALSE</code> ).
plot.optk	Logical value. If TRUE, this function plots the heuristic measure versus the number of clusters (default value is <code>plot.optk = TRUE</code> )
ret.optk	Logical value. For large datasets this computation can take a while and it might be interesting to save it. If TRUE, the function returns a copy of the <i>bdm</i> instance with the values of S2NR attached as <i>bdm\$optk</i> (default value is <code>ret.optk = FALSE</code> ).
layer	The <i>bdm\$ptsne</i> layer to be used (default value is <code>layer = 1</code> ).

## Details

The underlying idea is that neighbouring clusters in the embedding correspond to close clusters in the high dimensional space, *i.e.* this merging heuristic is based on the spatial distribution of clusters. For each cluster (child cluster) we choose the neighboring cluster with steepest gradient along their common border (father cluster). Thus, we get a set of pairs of clusters (child/father) to be potentially merged. Given this set of candidates, the merging is performed recursively choosing, at each step, the pair of child/father clusters that results in a minimum loss of S2NR. Typically some clusters dominate over all of their neighboring clusters. These clusters have no *father*. Thus, once all possible mergings have been performed we reach a *blocked* state where only the dominant clusters remain. This situation identifies a hierarchy level in the clustering. When this situation is reached, the algorithm starts a new merging round, identifying the child/father relations at that level of the hierarchy. The process stops when only two clusters remain. Usually, the clustering hierarchy is clearly depicted by singular points in the S2NR function. This is a hint that the low dimensional clustering configuration is an image of a hierarchycal configuration in the high dimensional space. See `bdm.optk.plot()`.



**Value**

None if `ret.optk = FALSE`. Else, a copy of the input *bdm* instance with new element *bdm\$optk* (a matrix).

**Examples**

```
# --- load mapped dataset
bdm.example()
# --- compute optimal number of clusters and attach the computation
bdm.optk.s2nr(ex$map, data = ex$data, plot.optk = TRUE, ret.optk = FALSE)
```

bdm.pakde

*Perplexity-adaptive kernel density estimation***Description**

Starts the paKDE algorithm (second step of the mapping protocol).

**Usage**

```
bdm.pakde(
  bdm,
  ppx = 100,
  g = 200,
  g.exp = 3,
  mpi.cl = NULL,
  threads = 2,
  layer = 1
)
```

**Arguments**

<code>bdm</code>	A <i>bdm</i> data mapping instance.
<code>ppx</code>	The value of perplexity to compute similarities in the low-dimensional embedding. Default value is <i>ppx = 100</i> .
<code>g</code>	The resolution of the density space grid ( $g * g$ cells). Default value is $g = 200$ .
<code>g.exp</code>	A numeric factor to avoid border effects. The grid limits will be expanded so as to enclose the density of the kernel of the most extreme embedded datapoints up to <i>g.exp</i> times $\sigma$ . Default value is <i>g.exp = 3</i> , i.e. the grid limits are expanded so as to enclose the 0.9986 of the probability mass of the most extreme kernels.
<code>mpi.cl</code>	An MPI (inter-node parallelization) cluster as returned by <i>bdm.mpi.start()</i> . Default value is <i>mpi.cl = NULL</i> , i.e. a 'SOCK' (intra-node parallelization) cluster is automatically generated.
<code>threads</code>	Number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory). Default value is <i>threads = 4</i> .
<code>layer</code>	The tSNE output layer. Default value is <i>layer = 1</i> .

## Details

When computing the *paKDE* the embedding area is discretized as a grid of size  $g \times g$  cells. In order to avoid border effects, the limits of the grid are expanded by default so as to enclose at least the 0.9986 of the cumulative distribution function ( $3\sigma$ ) of the kernels of the most extreme mapped points in each direction.

The presence of outliers in the embedding can lead to undesired expansion of the grid limits. We can overcome this using lower values of  $g.exp$ . By setting  $g.exp = 0$  the grid limits will be equal to the range of the embedding.

The values  $g.exp = c(1, 2, 3, 4, 5, 6)$  enclose cdf values of 0.8413, 0.9772, 0.9986, 0.99996, 0.99999, 1.0 respectively.

## Value

A copy of the input *bdm* instance with new element *bdm\$pakde* (paKDE output). *bdm\$pakde[[layer]]\$layer* = 'NC' stands for not computed layers.

## Examples

```
# --- load mapped dataset
bdm.example()
# --- run paKDE
## Not run:
m <- bdml.pakde(ex$map, ppx = 200, g = 200, g.exp = 3, threads = 4)
# --- plot paKDE output
bdml.pakde.plot(m)

## End(Not run)
```

---

bdml.pakde.plot	<i>Plot paKDE (density landscape)</i>
-----------------	---------------------------------------

---

## Description

Plot paKDE (density landscape)

## Usage

```
bdml.pakde.plot(bdm, pakde.pltt = NULL, pakde.lvls = 16, layer = 1)
```

## Arguments

bdm	A <i>bdm</i> instance as generated by <i>bdml.init()</i> or a list of them to make a comparative plot.
pakde.pltt	A colour palette to show levels in the paKDE plot. By default (pakde.pltt = NULL) the default palette is used.
pakde.lvls	The number of levels of the density heat-map (16 by default).
layer	The <i>bdm\$ptsne</i> layer to be used (default value is layer = 1).

**Value**

None.

**Examples**

```
bdm.example()
m <- bdm.pakde.plot(ex$map)
```

---

bdm.pMap

*Precision map (quantile map of betas)*


---

**Description**

Precision map (quantile map of betas)

**Usage**

```
bdm.pMap(
  bdm,
  pMap.levels = 8,
  pMap.cex = 0.1,
  pMap.bg = "#000000",
  colorbar = T
)
```

**Arguments**

bdm	A <i>bdm</i> instance as generated by <code>bdm.init()</code> .
pMap.levels	The number of levels of the quantile-map (8 by default).
pMap.cex	The size of the data-points (as in <code>par()</code> ). Default value is <code>ptsne.cex = 0.1</code> .
pMap.bg	The background colour of the qMap plot. Default value is <code>ptsne.bg = #FFFFFF</code> (white).
colorbar	A logical value (TRUE by default). FALSE hides the side colorbar.

**Value**

None.

**Examples**

```
bdm.example()
bdm.pMap(ex$map)
```

bdm.ptsne

*Parallelized t-SNE (ptSNE)***Description**

Starts the parallelized t-SNE algorithm (pt-SNE). This is the first step of the mapping protocol.

**Usage**

```
bdm.ptsne(
    data,
    bdm,
    theta = 0.5,
    Y.init = NULL,
    mpi.cl = NULL,
    threads = 4,
    layers = 2,
    info = 0
)
```

**Arguments**

data	Input data (a matrix, a big.matrix or a .csv file name).
bdm	A <i>bdm</i> data mapping instance.
theta	Accuracy/speed trade-off factor, a value between 0.33 and 0.8. (Default value is <i>theta</i> = 0.0). If <i>theta</i> < 0.33 the algorithm uses the exact computation of the gradient. The closer is this value to 1 the faster is the computation but the coarser is the approximation of the gradient.
Y.init	A $n * 2 * \text{layers}$ matrix with initial mapping positions. (By default <i>Y.init</i> =NULL will use random initial positions).
mpi.cl	MPI (inter-node parallelization) cluster as generated by <i>bdm.mpi.start()</i> . (By default <i>mpi.cl</i> = NULL a 'SOCK' (intra-node parallelization) cluster is generated).
threads	Number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory. Default value is <i>threads</i> = 4).
layers	Number of layers ( <i>minimum</i> 2, <i>maximum</i> the number of threads). Default value is <i>layers</i> = 2.
info	Output information: 1 yields inter-round results, 0 disables intermediate results. Default value is <i>info</i> = 0.

**Value**

A *bdm* data mapping instance.

## Examples

```
# --- load example dataset
bdm.example()
# --- perform ptSNE
## Not run:
# --- run ptSNE
m <- bdms.ptsne(ex$data, ex$map, threads = 10, layers = 2)
# --- plot the Cost function
bdm.cost(m)
# --- plot ptSNE output
bdm.ptsne.plot(m, class.lbls = ex$labels)

## End(Not run)
```

---

bdm.ptsne.plot	<i>Plot ptSNE (low-dimensional embedding)</i>
----------------	---

---

## Description

Plot ptSNE (low-dimensional embedding)

## Usage

```
bdm.ptsne.plot(
  bdms,
  ptsne.cex = 0.5,
  ptsne.bg = "#FFFFFF",
  class.lbls = NULL,
  class.pltt = NULL,
  layer = 1
)
```

## Arguments

bdms	A <i>bdms</i> instance as generated by <code>bdms.init()</code> or a list of them to make a comparative plot.
ptsne.cex	The size of the mapped data-points in the ptSNE plot. Default value is <code>ptsne.cex = 0.5</code> .
ptsne.bg	The background colour of the ptSNE plot. Default value is <code>ptsne.bg = #FFFFFF</code> (white).
class.lbls	A vector of numeric class labels. If <code>is.null(class.lbls)</code> and <code>!is.null(bdms\$wtt)</code> cluster labels are used.
class.pltt	A colour palette to show the class labels in the ptSNE plot. If <code>ptsne.pltt = NULL</code> (default value) the default palette is used.
layer	The <i>bdms\$ptsne</i> layer to be used (default value is <code>layer = 1</code> ).

**Value**

None.

**Examples**

```
bdm.example()
m <- bdmsne.plot(ex$map, class.lbls = ex$labels)
```

---

bdm.qMap	<i>ptSNE quantile-maps</i>
----------	----------------------------

---

**Description**

Maps quantitative variables onto the embedding space.

**Usage**

```
bdm.qMap(
  bdms,
  data,
  labels = NULL,
  subset = NULL,
  qMap.levels = 8,
  qMap.cex = 0.3,
  qMap.bg = "#FFFFFF",
  class.pltt = NULL,
  qtitle = NULL,
  cex.main = 1,
  colorbar = T,
  layer = 1
)
```

**Arguments**

bdms	A <i>bdms</i> instance as generated by <code>bdms.init()</code> .
data	A matrix/data.frame to be mapped.
labels	A length <code>nrow(bdms\$data)</code> vector of class labels to overlay onto the embedding. Label values are factorized as <code>as.numeric(as.factor(labels))</code> . Default value is <code>labels = NULL</code> .
subset	A numeric vector with the indexes of a subset of data. Data-points in the subset are heat-mapped and the rest are shown in light grey. By default all data-points are heat-mapped.
qMap.levels	The number of levels of the quantile-map (8 by default).
qMap.cex	The size of the data-points (as in <code>par()</code> ).

qMap.bg	The background colour of the qMap plot. Default value is ptsne.bg = #FFFFFF (white).
class.pltt	If !is.null(labels) or !is.null(bdm\$lbls), a colour palette to show class labels with the qMap plots. By default (qMap.pltt = NULL) the default palette is used.
qtitle	A vector of strings with titles for the plots. Default value is qtitle=NULL.
cex.main	The font size of the title (as in par()).
colorbar	A logical value (TRUE by default). FALSE hides the side colorbar.
layer	The number of a layer (1 by default).

### Details

This is not a heat-map but a quantile-map plot. This function splits the range of each variable into as many quantiles as specified by *levels* so that the color gradient will hardly ever correspond to a constant numeric gradient. Thus, the mapping will show more evenly distributed colors though at the expense of possibly exaggerating artifacts. For variables with very extrem distributions, it will be impossible to find as many quantiles as desired and the distribution of colors will not be so homogeneous.

### Value

None.

### Examples

```
bdm.example()
bdm.qMap(ex$map, ex$data)
# --- show only components (1, 2, 4, 8) of the GMM
bdm.qMap(ex$map, ex$data, subset = which(ex$map$lbls %in% c(1, 4, 8, 16)))
```

---

bdm.restart	<i>Restart pt-SNE</i>
-------------	-----------------------

---

### Description

Restarts the ptSNE algorithm (runs more epochs).

### Usage

```
bdm.restart(
  data,
  bdm,
  epochs = NULL,
  iters = NULL,
  mpi.cl = NULL,
  threads = NULL,
  layers = NULL,
  info = 0
)
```

**Arguments**

data	Input data (a matrix, a big.matrix or a .csv file name).
bdm	A <i>bdm</i> data mapping instance.
epochs	Number of epochs to run. Default value <i>epochs</i> = <i>NULL</i> runs $4 * \log(n)$ epochs.
iters	Number of iters per epoch. Default value <i>iters</i> = <i>NULL</i> runs $4 * \log(\text{thread\_size})$ iters/epoch.
mpi.cl	An MPI (inter-node parallelization) cluster as returned by <i>bdm.mpi.start()</i> . Default value is <i>mpi.cl</i> = <i>NULL</i> , i.e. a 'SOCK' (intra-node parallelization) cluster is automatically generated.
threads	Number of parallel threads (according to data size and hardware resources, i.e. number of cores and available memory). Default value is <i>threads</i> = 4.
layers	Number of layers ( <i>minimum</i> 2, <i>maximum</i> the number of threads). Default value is <i>layers</i> = 2.
info	Output information: 1 yields inter-round results, 0 disables intermediate results. Default value is 0.

**Value**

A *bdm* data mapping instance.

**Examples**

```
# --- load example dataset
bdm.example()
## Not run:
# --- restart ptSNE
m <- bdm.restart(ex$data, ex$map, epochs = 50)

## End(Not run)
```

---

bdm.wtt

*Watertrack transform (WTT)*


---

**Description**

Starts the WTT algorithm (third setp of the mapping protocol).

**Usage**

```
bdm.wtt(bdm, layer = 1)
```

**Arguments**

bdm	A <i>bdm</i> data mapping instance.
layer	The ptSNE output layer. Default value is <i>layer</i> = 1.



## Details

This function requires the up-stream step *bdm.pakde()*.

## Value

A *bdm* data mapping instance.

## Examples

```
# --- load mapped dataset
bdm.example()
# --- perform WTT
m <- bdm.wtt(ex$map)
# --- plot WTT output
bdm.wtt.plot(m)
```

---

bdm.wtt.plot	<i>Plot WTT (clustering)</i>
--------------	------------------------------

---

## Description

Plot WTT (clustering)

## Usage

```
bdm.wtt.plot(
  bdm,
  pakde.pltt = NULL,
  pakde.lvls = 16,
  wtt.lwd = 1,
  plot.peaks = T,
  labels.cex = 1,
  layer = 1
)
```

## Arguments

<i>bdm</i>	A <i>bdm</i> instance as generated by <i>bdm.init()</i> or a list of them to make a comparative plot.
<i>pakde.pltt</i>	A colour palette to show levels in the paKDE plot. By default ( <i>pakde.pltt</i> = <i>NULL</i> ) the default palette is used.
<i>pakde.lvls</i>	The number of levels of the density heat-map (16 by default).
<i>wtt.lwd</i>	The width of the watertrack lines (as set in <i>par()</i> ).
<i>plot.peaks</i>	Logical value (TRUE by default). If set to TRUE and the up-stream step <i>bdm\$wtt()</i> is computed marks the peak of each cluster.
<i>labels.cex</i>	If <i>plot.peaks</i> is TRUE, the size of the labels of the clusters (as set in <i>par()</i> ). By default <i>labels.cex</i> = 0.0 and the labels of the clusters are not depicted.
<i>layer</i>	The <i>bdm\$ptsne</i> layer to be used (default value is <i>layer</i> = 1).

**Value**

None.

**Examples**

```
bdm.example()  
m <- bdm.wtt.plot(ex$map)
```

# Index

`bdm.bboxp`, [2](#)  
`bdm.cost`, [3](#)  
`bdm.dMap`, [4](#)  
`bdm.dMap.plot`, [5](#)  
`bdm.example`, [7](#)  
`bdm.hlCorr`, [7](#)  
`bdm.init`, [8](#)  
`bdm.knp`, [9](#)  
`bdm.knp.plot`, [10](#)  
`bdm.labels`, [11](#)  
`bdm.merge.s2nr`, [12](#)  
`bdm.mpi.start`, [13](#)  
`bdm.mpi.stop`, [13](#)  
`bdm.mtsne`, [14](#)  
`bdm.optk.plot`, [15](#)  
`bdm.optk.s2nr`, [16](#)  
`bdm.pakde`, [17](#)  
`bdm.pakde.plot`, [18](#)  
`bdm.pMap`, [19](#)  
`bdm.ptsne`, [20](#)  
`bdm.ptsne.plot`, [21](#)  
`bdm.qMap`, [22](#)  
`bdm.restart`, [23](#)  
`bdm.wtt`, [24](#)  
`bdm.wtt.plot`, [25](#)