Package 'mbkmeans'

July 13, 2025

Type Package

Title Mini-batch K-means Clustering for Single-Cell RNA-seq

Version 1.24.0

Description Implements the mini-batch k-means algorithm for large datasets, including support for on-disk data representation.

Depends R (>= 3.6)

Imports methods, DelayedArray, Rcpp, S4Vectors, SingleCellExperiment, SummarizedExperiment, ClusterR, benchmarkme, Matrix, BiocParallel

Suggests beachmat, HDF5Array, Rhdf5lib, BiocStyle, TENxPBMCData, scater, DelayedMatrixStats, bluster, knitr, testthat, rmarkdown

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

LinkingTo Rcpp, RcppArmadillo (>= 0.7.2), Rhdf5lib, beachmat, ClusterR

SystemRequirements C++11

VignetteBuilder knitr

biocViews Clustering, GeneExpression, RNASeq, Software, Transcriptomics, Sequencing, SingleCell

BugReports https://github.com/drisso/mbkmeans/issues

git_url https://git.bioconductor.org/packages/mbkmeans

git_branch RELEASE_3_21

git_last_commit bffcc1e

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-07-13

Author Yuwei Ni [aut, cph],
Davide Risso [aut, cre, cph],
Stephanie Hicks [aut, cph],
Elizabeth Purdom [aut, cph]

Maintainer Davide Risso <risso.davide@gmail.com>

Contents

`blocksize` *blocksize*

Description

Return the maximum number of rows to use based on the amount of ram memory.

Usage

```
blocksize(data, ram = get_ram())
```

Arguments

`data`	matrix-like object.
`ram`	the max amount of ram (in bytes) to use.

Value

Numeric value of the maximum number of rows.

Examples

```
data <- matrix(NA, nrow = 100, ncol=1000)
blocksize(data, ram=1e6)
```

clusterRows *Cluster rows of a matrix*

Description

Cluster rows of a matrix-like object with a variety of algorithms.

Details

This function is deprecated. Please use the `clusterRows` function in the `bluster` Bioconductor package.

compute_wcss *Compute Whithin-Cluster Sum of Squares*

Description

Given a vector of cluster labels, a matrix of centroids, and a dataset, it computes the WCSS.

Usage

```
compute_wcss(clusters, cent, data)
```

Arguments

`clusters`	numeric vector with the cluster assignments.
`cent`	numeric matrix with the centroids (clusters in rows, variables in columns).
`data`	matrix-like object containing the data (numeric or integer).

Value

A numeric vector with the value of WCSS per cluster.

Examples

```
data = matrix(1:30,nrow = 10)
cl <- mini_batch(data, 2, 10, 10)
compute_wcss(cl$Clusters, cl$centroids, data)
```

mbkmeans *Mini-Batch k-means for large single cell sequencing data*

Description

This is an implementation of the mini-batch k-means algorithm of Sculley (2010) for large single cell sequencing data with the dimensionality reduction results as input in the reducedDim() slot.

Usage

```
mbkmeans(x, ...)

## S4 method for signature 'SummarizedExperiment'
mbkmeans(x, whichAssay = 1, ...)

## S4 method for signature 'SingleCellExperiment'
mbkmeans(x, reduceMethod = "PCA", whichAssay = 1, ...)

## S4 method for signature 'LinearEmbeddingMatrix'
mbkmeans(x, ...)

## S4 method for signature 'ANY'
mbkmeans(
  x,
  clusters,
  batch_size = min(500, NCOL(x)),
  max_iters = 100,
  num_init = 1,
  init_fraction = batch_size/NCOL(x),
  initializer = "kmeans++",
  compute_labels = TRUE,
  calc_wcss = FALSE,
  early_stop_iter = 10,
  verbose = FALSE,
  CENTROIDS = NULL,
  tol = 1e-04,
  BPPARAM = BiocParallel::SerialParam(),
  ...
)
```

Arguments

x	The object on which to run mini-batch k-means. It can be a matrix-like object (e.g., matrix, Matrix, DelayedMatrix, HDF5Matrix) with genes in the rows and samples in the columns. Specialized methods are defined for SummarizedExperiment and SingleCellExperiment.
...	passed to ‘blockApply‘.

whichAssay	The assay to use as input to mini-batch k-means. If x is a SingleCellExperiment, this is ignored unless reduceMethod = NA.
reduceMethod	Name of dimensionality reduction results to use as input to mini-batch k-means. Set to NA to use the full matrix.
clusters	the number of clusters
batch_size	the size of the mini batches. By default, it equals the minimum between the number of observations and 500.
max_iters	the maximum number of clustering iterations
num_init	number of times the algorithm will be run with different centroid seeds
init_fraction	proportion of data to use for the initialization centroids (applies if initializer is *kmeans++*). Should be a float number between 0.0 and 1.0. By default, it uses the relative batch size.
initializer	the method of initialization. One of *kmeans++* and *random*. See details for more information
compute_labels	logcical indicating whether to compute the final cluster labels.
calc_wcss	logical indicating whether the per-cluster WCSS is computed. Ignored if 'compute_labels = FALSE'.
early_stop_iter	continue that many iterations after calculation of the best within-cluster-sum-of-squared-error
verbose	either TRUE or FALSE, indicating whether progress is printed during clustering
CENTROIDS	a matrix of initial cluster centroids. The rows of the CENTROIDS matrix should be equal to the number of clusters and the columns should be equal to the columns of the data
tol	a float number. If, in case of an iteration (iteration > 1 and iteration < max_iters) 'tol' is greater than the squared norm of the centroids, then kmeans has converged
BPPARAM	See the 'BiocParallel' package. Only the label assignment is done in parallel.

Details

The implementation is largely based on the MiniBatchKmeans function of the ClusterR package. The contribution of this package is to provide support for on-disk data representations such as HDF5, through the use of DelayedMatrix and HDF5Matrix objects, as well as for sparse data representation through the classes of the Matrix package. We also provide high-level methods for objects of class SummarizedExperiment, SingleCellExperiment, and LinearEmbeddingMatrix.

This function performs k-means clustering using mini batches.

kmeans++: kmeans++ initialization. Reference : http://theory.stanford.edu/~sergei/papers/kMeansPP-soda.pdf AND http://stackoverflow.com/questions/5466323/how-exactly-does-k-means-work

random: random selection of data rows as initial centroids

Value

A list with the following attributes: centroids, WCSS_per_cluster, best_initialization, iters_per_initialization.

a list with the following attributes: centroids, WCSS_per_cluster, best_initialization, iters_per_initialization

Author(s)

Lampros Mouselimis and Yuwei Ni

References

Sculley. Web-Scale K-Means Clustering. WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA. ACM 978-1-60558-799-8/10/04.

https://github.com/mlampros/ClusterR

Examples

```
library(SummarizedExperiment)
se <- SummarizedExperiment(matrix(rnorm(100), ncol=10))
mbkmeans(se, clusters = 2)
library(SingleCellExperiment)
sce <- SingleCellExperiment(matrix(rnorm(100), ncol=10))
mbkmeans(sce, clusters = 2, reduceMethod = NA)
x<-matrix(rnorm(100), ncol=10)
mbkmeans(x,clusters = 3)
```

`MbkmeansParam` *Mini-batch k-means clustering*

Description

Run the mini-batch k-means `mbkmeans` function with the specified number of centers within `clusterRows` from the `bluster` Bioconductor package.

Usage

```
MbkmeansParam(centers, ...)
```

Arguments

`centers`	An integer scalar specifying the number of centers. Alternatively, a function that takes the number of observations and returns the number of centers. Note, the `mbkmeans` function uses the argument `clusters` argument to represent this argument. However, we use `centers` to match
`...`	Further arguments to pass to `mbkmeans`.

Details

This function is deprecated. Please use the `MbkmeansParam` function in the `bluster` Bioconductor package.

mini_batch *Mini_batch*

Description

Mini-batch-k-means for matrix-like objects

Usage

```
mini_batch(
  data,
  clusters,
  batch_size,
  max_iters,
  num_init = 1L,
  init_fraction = 1,
  initializer = "kmeans++",
  compute_labels = TRUE,
  calc_wcss = FALSE,
  early_stop_iter = 10L,
  verbose = FALSE,
  CENTROIDS = NULL,
  tol = 1e-04
)
```

Arguments

data	numeric or integer matrix-like object.
clusters	the number of clusters.
batch_size	the size of the mini batches.
max_iters	the maximum number of clustering iterations.
num_init	number of times the algorithm will be run with different centroid seeds.
init_fraction	percentage of data to use for the initialization centroids (applies if initializer is *kmeans++*). Should be a float number between 0.0 and 1.0.
initializer	the method of initialization. One of *kmeans++* and *random*. See details for more information.
compute_labels	logical indicating whether to compute the final cluster labels.
calc_wcss	logical indicating whether the within-cluster sum of squares should be computed and returned (ignored if 'compute_labels = FALSE').
early_stop_iter	continue that many iterations after calculation of the best within-cluster-sum-of-squared-error.
verbose	logical indicating whether progress is printed on screen.

CENTROIDS	an optional matrix of initial cluster centroids. The rows of the CENTROIDS matrix should be equal to the number of clusters and the columns should be equal to the columns of the data.
tol	convergence tolerance.

Details

This function performs k-means clustering using mini batches. It was inspired by the implementation in https://github.com/mlampros/ClusterR.

The input matrix can be in any format supported by the 'DelayedArray' / 'beachmat' framework, including the matrix classes defined in the 'Matrix' package and the 'HDFMatrix' class.

There are two possible initializations.

kmeans++: kmeans++ initialization.

random: random selection of data rows as initial centroids.

Value

a list with the following attributes:

centroids: the final centroids;

WCSS_per_cluster (optional): the final per-cluster WCSS.

best_initialization: which initialization value led to the best WCSS solution;

iters_per_initialization: number of iterations per each initialization;

Clusters (optional): the final cluster labels.

References

Sculley, D., 2010, April. Web-scale k-means clustering. In Proceedings of the 19th international conference on World wide web (pp. 1177-1178). ACM.

Arthur, D. and Vassilvitskii, S., 2007, January. k-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (pp. 1027-1035). Society for Industrial and Applied Mathematics.

Examples

```
data = matrix(1:30,nrow = 10)
mini_batch(data, 2, 10, 10)
```

predict_mini_batch *Predict_mini_batch*

Description

Prediction function for mini-batch k-means applied to matrix-like objects.

Usage

```
predict_mini_batch(data, CENTROIDS)
```

Arguments

data	matrix-like objectcontaining numeric or integer data (obseravtions in rows, variables in columns).
CENTROIDS	a matrix of initial cluster centroids. The rows of the CENTROIDS matrix should be equal to the number of clusters and the columns should equal the columns of the data.

Details

This function takes the data and the output centroids and returns the clusters.

This implementation relies very heavily on the MiniBatchKmeans implementation. We provide the ability to work with other matrix-like objects other than base matrices (e.g, DelayedMatrix and HDF5Matrix) through the beachmat library.

Value

it returns a vector with the clusters.

Author(s)

Yuwei Ni

Examples

```
data(iris)
km = mini_batch(as.matrix(iris[,1:4]), clusters = 3,
               batch_size = 10, max_iters = 10)
clusters = predict_mini_batch(as.matrix(iris[,1:4]),
                              CENTROIDS = km$centroids)
```

predict_mini_batch_r *Compute labels for mini-batch k-means*

Description

Given a data matrix and a centroid matrix, it assigns each data point to the closest centroid, using block processing.

Usage

```
predict_mini_batch_r(
  data,
  centroids,
  BPPARAM = BiocParallel::SerialParam(),
  ...
)
```

Arguments

data	a matrix-like object with features in row and samples in columns.
centroids	a matrix with the coordinates of the centroids.
BPPARAM	for parallel computations. See the 'BiocParallel' package.
...	passed to 'blockApply'.

Value

a vector of cluster labels for each observation.

Examples

```
data(iris)
km <- mini_batch(as.matrix(iris[,1:4]), clusters = 3,
                 batch_size = 10, max_iters = 100)
predict_mini_batch_r(t(as.matrix(iris[,1:4])), km$centroids)
```

Index