Package: Rsomoclu (via r-universe)

October 5, 2026

Version 1.7.7

Title Somoclu

Imports kohonen

Description Somoclu is a massively parallel implementation of self-organizing maps. It exploits multicore CPUs and it can be accelerated by CUDA. The topology of the map can be planar or toroid and the grid of neurons can be rectangular or hexagonal . Details refer to (Peter Wittek, et al (2017)) <doi:10.18637/jss.v078.i09>.

URL https://peterwittek.github.io/somoclu/

BugReports https://github.com/peterwittek/somoclu/issues

License MIT + file LICENSE

Author Peter Wittek [aut], Shichao Gao [cre]

Maintainer Shichao Gao <xgdgsc@gmail.com>

NeedsCompilation yes

SystemRequirements GNU make

LinkingTo Rcpp

LazyData true

Config/pak/sysreqs make

Repository https://cran.r-universe.dev

Date/Publication 2025-12-20 14:50:02 UTC

RemoteUrl https://github.com/cran/Rsomoclu

RemoteRef HEAD

RemoteSha c6e482d664ab8d4cac05f3f998168a985a3d4d74

Contents

rgbs 2
Rsomoclu.kohonen 2
Rsomoclu.train 3

Index **6**

`rgbs` *tiny rgbs data*

Description

tiny rgbs data for testing

Usage

```
rgbs
```

Format

matrix in plain text form

`Rsomoclu.kohonen` *convert Somoclu train result to kohonen class for plotting*

Description

A function call to convert Somoclu train result to kohonen class for plotting.

Usage

```
Rsomoclu.kohonen(input_data, result, n.hood = NULL, toroidal = FALSE)
```

Arguments

`input_data`	input data, matrix format
`result`	The result returned by Rsomoclu.train
`n.hood`	Same as in koohonen, the shape of the neighbourhood, either "circular" or "square". The latter is the default for rectangular maps, the former for hexagonal maps.
`toroidal`	if TRUE, the edges of the map are joined. Note that in a hexagonal toroidal map, the number of rows must be even.

Value

An object of class kohonen for plotting.

See Also

https://www.r-bloggers.com/2014/02/self-organising-maps-for-customer-segmentation-using-r/

Examples

```
library('Rsomoclu')
library('kohonen')
data("rgbs", package = "Rsomoclu")
input_data <- rgbs
input_data <- data.matrix(input_data)
nSomX <- 20
nSomY <- 20
nEpoch <- 10
radius0 <- 0
radiusN <- 0
radiusCooling <- "linear"
scale0 <- 0
scaleN <- 0.01
scaleCooling <- "linear"
kernelType <- 0
mapType <- "planar"
gridType <- "rectangular"
compactSupport <- FALSE
codebook <- NULL
neighborhood <- "gaussian"
stdCoeff <- 0.5
res <- Rsomoclu.train(input_data, nEpoch, nSomX, nSomY,
                      radius0, radiusN,
                      radiusCooling, scale0, scaleN,
                      scaleCooling,
                      kernelType, mapType, gridType, compactSupport,
                      neighborhood, stdCoeff, codebook)
## Convert to kohonen object for plotting
sommap = Rsomoclu.kohonen(input_data, res)
## Show 'codebook'
plot(sommap, type="codes", main = "Codes")
## Show 'component planes'
plot(sommap, type = "property", property = sommap$codes[[1]][,1],
     main = colnames(sommap$codes)[1])
## Show 'U-Matrix'
plot(sommap, type="dist.neighbours")
```

`Rsomoclu.train` *Train function for Somoclu*

Description

A function call to Somoclu to train the Self Organizing Map.

Usage

```
Rsomoclu.train(input_data, nEpoch, nSomX, nSomY,
                      radius0, radiusN,
```

```
radiusCooling, scale0, scaleN,
scaleCooling,
kernelType, mapType, gridType, compactSupport,
neighborhood, stdCoeff, codebook, vectDistance)
```

Arguments

input_data	input data, matrix format
nEpoch	Maximum number of epochs
nSomX	Number of columns in map (size of SOM in direction x)
nSomY	Number of rows in map (size of SOM in direction y)
radius0	Start radius (default: half of the map in direction min(x,y))
radiusN	End radius (default: 1)
radiusCooling	Radius cooling strategy: linear or exponential (default: linear)
scale0	Starting learning rate (default: 1.0)
scaleN	Finishing learning rate (default: 0.01)
scaleCooling	Learning rate cooling strategy: linear or exponential (default: linear)
kernelType	Kernel type 0: Dense CPU 1: Dense GPU 2: Sparse CPU (default: 0)
mapType	Map type: planar or toroid (default: "planar")
gridType	Grid type: square or hexagonal (default: "rectangular")
compactSupport	Compact support for Gaussian neighborhood, (default:TRUE)
neighborhood	Neighborhood function: gaussian or bubble (default: "gaussian")
codebook	initial codebook, (default:NULL)
stdCoeff	The coefficient in the Gaussian neighborhood function exp(-||x-y||^2/(2*(coeff*radius)^2)), (default:0.5)
vectDistance	the vector distance function "norm-3", "norm-6", "norm-2"(same as default) "norm-inf", is supported with kerneltype = 0 only , (default:euclidean)

Value

a list including elements

codebook	the codebook
globalBmus	global Best Matching Unit matrix
uMatrix	uMatrix

Author(s)

Peter Wittek, Shichao Gao

References

Peter Wittek, Shi Chao Gao, Ik Soo Lim, Li Zhao (2017). Somoclu: An Efficient Parallel Library for Self-Organizing Maps. Journal of Statistical Software, 78(9), 1-21. doi:10.18637/jss.v078.i09.

Examples

```
library('Rsomoclu')
data("rgbs", package = "Rsomoclu")
input_data <- rgbs
input_data <- data.matrix(input_data)
nSomX <- 10
nSomY <- 10
nEpoch <- 10
radius0 <- 0
radiusN <- 0
radiusCooling <- "linear"
scale0 <- 0
scaleN <- 0.01
scaleCooling <- "linear"
kernelType <- 0
mapType <- "planar"
gridType <- "rectangular"
compactSupport <- FALSE
codebook <- NULL
neighborhood <- "gaussian"
stdCoeff <- 0.5
vectDistance <- "euclidean"
res <- Rsomoclu.train(input_data, nEpoch, nSomX, nSomY,
                      radius0, radiusN,
                      radiusCooling, scale0, scaleN,
                      scaleCooling,
                      kernelType, mapType, gridType, compactSupport, neighborhood,
                      stdCoeff, codebook, vectDistance)
res$codebook
res$globalBmus
res$uMatrix
library('kohonen')
sommap = Rsomoclu.kohonen(input_data, res)
```

Index

∗ **datasets**
rgbs, 2

rgbs, 2
Rsomoclu.kohonen, 2
Rsomoclu.train, 3