Package: GloScope (via r-universe)

November 25, 2026

Type Package

Title Population-level Representation on scRNA-Seq data

Version 2.4.0

Description This package aims at representing and summarizing the entire single-cell profile of a sample. It allows researchers to perform important bioinformatic analyses at the sample-level such as visualization and quality control. The main functions Estimate sample distribution and calculate statistical divergence among samples, and visualize the distance matrix through MDS plots.

BugReports https://github.com/epurdom/GloScope/issues

License Artistic-2.0

Encoding UTF-8

Imports utils, stats, MASS, mclust, ggplot2, RANN, FNN, BiocParallel, mvnfast, SingleCellExperiment, rlang, RColorBrewer, pheatmap, vegan, cluster, boot, permute

Depends R (>= 4.4.0)

Suggests BiocStyle, testthat (>= 3.0.0), knitr, rmarkdown, zellkonverter

VignetteBuilder knitr

LazyData false

biocViews DataRepresentation, QualityControl, RNASeq, Sequencing, Software, SingleCell

RoxygenNote 7.3.3

Config/testthat/edition 3

Config/pak/sysreqs zlib1g-dev

Repository Bioconductor 3.24

Date/Publication 2026-11-23 22:51:14 UTC

RemoteUrl https://github.com/bioc/GloScope

RemoteRef RELEASE_3_24

RemoteSha 44cead1299883ac6cf36ba85d5f94682e23cc85c

Contents

example_SCE 2
getMetrics 3
gloscope 6
gloscopeProp 7
plotCI 8
plotHeatmap 9
plotMDS 11

Index **13**

`example_SCE` *SingleCellExperiment containing example inputs to GloScope*

Description

'example_SCE' is a SingleCellExperiment object which contains PCA embeddings and metadata for PBMCs from 20 COVID-infected and healthy control patients. Each sample is reduced to a random subset of 500 cells, for a total of 10,000 cells. The 'colData' slot of the object contains the metadata for each cell, its sample ID and phenotype. The dimensionality reductions slot contains the first 50 PCs, and these embeddings are provided by the authors of "Single-cell multi-omics analysis of the immune response in COVID-19" (Stephenson et al., 2021; doi: 10.1038/s41591-021-01329-2).

'example_SCE_small' is a SingleCellExperiment with the same structure as 'example_SCE', but only containing data from the first five samples. This is a smaller set for examples.

Format

A SingleCellExperiment object with metadata and PCA embeddings

Value

A SingleCellExperiment object

Examples

```
# Code to create the small SCE from the full sample
# Reduction to 5 samples demonstrates data extraction from SCE objects
data(example_SCE)
sample_ids <- SingleCellExperiment::colData(example_SCE)$sample_id
whKeep <- which(sample_ids %in% unique(sample_ids)[seq_len(5)])
example_SCE_small <- SingleCellExperiment::SingleCellExperiment(
 assays=list(counts=matrix(rep(0,2500),ncol=2500)),
 colData=SingleCellExperiment::colData(example_SCE)[whKeep,],
 reducedDims=list("PCA"=SingleCellExperiment::reducedDim(example_SCE,"PCA")[whKeep,]))
```

getMetrics *Calculate metrics on separation in distance matrix due to grouping variable*

Description

These functions are wrappers for calculating common metrics for the amount of separation in a distance matrix due to a grouping (factor) variable and creating bootstrap confidence intervals and permutation tests.

Usage

```
getMetrics(
  dist_mat,
  metadata_df,
  metrics = c("anosim", "adonis2", "silhouette"),
  sample_id,
  group_vars,
  checkData = TRUE,
  permuteTest = FALSE,
  permutations = 100
)

bootCI(
  dist_mat,
  metadata_df,
  metrics = "anosim",
  sample_id,
  group_vars,
  R = 1000,
  ci_type = c("perc", "norm", "basic", "stud", "bca"),
  ci_conf = 0.95,
  ...
)

bootGloscope(
  dist_mat,
  metadata_df,
  metrics = "anosim",
  sample_id,
  group_vars,
  R = 1000,
  ...
)
```

Arguments

dist_mat	The divergence matrix output of ‘gloscope()‘. Should be a symmetric, square matrix. For ‘bootCI‘ the argument can be a list of distance matrices.
metadata_df	A data frame contains each sample’s metadata. Note this is NOT at the cell-level, and should have the same number of rows as dist_mat.
metrics	vector of statistics to calculate. For ‘bootstrap_gloscope‘ must be single value.
sample_id	The column name or index in metadata_df that contains the sample ID. This is for ensuring alignment between the dist_mat and the metadata_df. The row-names of dist_mat are expected to match the sample_id values.
group_vars	vector of names of grouping variables in metadata_df for which to calculate metrics. For ‘bootstrap_gloscope‘ must be single value.
checkData	Whether to check whether dist_mat, metadata_df, and sample_id match, for example in terms of dimensions and rownames. Mainly used internally.
permuteTest	whether to run permutation tests on each of the metrics
permutations	if ‘permuteTest=TRUE‘, an integer value defines the number of permutations. Can also except output of how for more fine control of the permutation mechanisms.
R	number of bootstrap replicates. See boot.
ci_type	Single character value. The type of confidence interval to compute. Passed to argument ‘type‘ in boot.ci
ci_conf	Scalar value between 0 and 1. The confidence level requested. Passed to argument ‘conf‘ in boot.ci.
...	arguments passed to boot

Details

The function ‘getMetrics‘ is a simple wrapper for calculating statistics that summarize the difference between distances within and between groupings. If the variable defined by group_var does not have at least two groupings, the function will return a NA.

The options "anosim" and "adonis2" are wrappers to the functions of that name in the package ‘vegan‘; we have turned off the permutation testing option of those functions. The functions in ‘vegan‘ have greater capability, and in particular adonis2 has capability to handle more complicated testing paradigms than a simple grouping factor. Permutation tests for these statistics are handled by the functions of ‘vegan‘.

The option "silhouette" calls the function of that name from the package ‘cluster‘ and calculates the silhouette width of each group and then averages them across groups. The permutation test is coded making use of the package ‘permute‘, similar to ‘vegan‘, so that control of the permutation mechanism is possible in the same way.

‘bootCI‘ is a wrapper function to boot.ci. ‘boot.ci‘ can be called directly on the output of ‘bootGloscope‘. The main advantage of ‘bootCI‘ is to calculate bootstrap CI over multiple choices of metrics, variables, and/or distance matrices. Unlike ‘boot.ci‘, ‘bootCI‘ does not allow different choices of confidence interval types or levels, so ‘ci_type‘ and ‘ci_level‘ must be of length 1. For this kind of multiplicity, call ‘boot.ci‘ directly on the output of ‘bootGloscope‘.

The function ‘bootGloscope‘ is a wrapper to the boot function for creating bootstraps of one of the metrics calculated by ‘getMetrics‘. Most users will probably prefer ‘bootCI‘

Value

'getMetrics' creates a data frame containing the statistic for each combination of metric and grouping variable with columns

- metric
- grouping
- statistic
- pval (if 'permuteTest=TRUE')

'bootCI' creates a data frame containing the statistic for each combination of metric and grouping variable with columns with the upper and lower bounds of the requested confidence intervals

- metric
- grouping
- statistic
- lower
- upper

'bootGloscope' returns an object of class 'boot' created by boot.

See Also

anosim, adonis2, silhouette, how

boot.ci

boot

Examples

```
data(example_SCE_small)
sample_ids <- SingleCellExperiment::colData(example_SCE_small)$sample_id
# Run gloscope on first 10 PCA embeddings
# We use 'KNN' option for speed ('GMM' is slightly slower)
pca_embeddings <- SingleCellExperiment::reducedDim(example_SCE_small,"PCA")
pca_embeddings_subset <- pca_embeddings[,seq_len(10)] # select the first 10 PCs
dist_result <- gloscope(pca_embeddings_subset, sample_ids,
   dens="KNN",
   BPPARAM = BiocParallel::SerialParam(RNGseed=2))
# make a per-sample metadata
sample_metadata <- as.data.frame(unique(SingleCellExperiment::colData(example_SCE_small)[,c(1,2)]))
# make another variable
sample_metadata$grouping<-c(rep(c("A","B"),each=2),"A")
getMetrics(dist_result,metadata_df=sample_metadata, sample_id="sample_id",
  group_vars="phenotype")
# run permutation tests:
getMetrics(dist_result,metadata_df=sample_metadata, sample_id="sample_id",
  group_vars=c("phenotype","grouping"), permuteTest=TRUE)

# calculate many bootstraps -- for speed up we set R ridiculously low
manyboot<-bootCI(list("Distance 1"=dist_result,"Another distance"=dist_result),
```

```
  sample_metadata,"sample_id",
  metrics=c("anosim","silhouette"),group_vars=c("phenotype","grouping"),R=20)

# single bootstrap of anosim
bootout<-bootGloscope(dist_result,sample_metadata,"sample_id",
  metric="anosim",group_var="phenotype")
#work with the boot object using functions in boot package:
library(boot)
print(bootout)
boot.ci(bootout)
```

---

gloscope *Calculate statistical divergence between all sample pairs*

Description

This function calculates a matrix of pairwise divergences between input samples of single cell data.

Usage

```
gloscope(
  embedding_matrix,
  cell_sample_ids,
  dens = c("GMM", "KNN"),
  dist_metric = c("KL", "JS"),
  r = 10000,
  num_components = c(5, 10, 15, 20),
  k = 50,
  GMM_params = list(modelNames = c("VVE"), verbose = FALSE, plot = FALSE),
  KNN_params = NULL,
  BPPARAM = BiocParallel::SerialParam(),
  prefit_density = NULL,
  return_density = FALSE
)
```

Arguments

embedding_matrix	a matrix or data.frame of latent embeddings with rows corresponding to cells and columns to dimensions
cell_sample_ids	a vector of the samples IDs each cell comes from. Length must match the number of rows in ‘embedding_matrix‘
dens	the density estimation. One of c("GMM","KNN")
dist_metric	distance metric to calculate the distance. One of c("KL","JS")
r	number of Monte Carlo simulations to generate

num_components	a vector of integers for the number of components to fit GMMs to, default is c(5,10,15,20)
k	number of nearest neighbours for KNN density estimation, default k = 50.
GMM_params	optional mclust parameters, default is to restrict the fit model to only VVE
KNN_params	optional arguments for either 'FNN:KL.dist' (KL) or 'RANN::nn2' (JS), default is NULL
BPPARAM	BiocParallel parameters, default is running in serial. Set random seed with 'RNGseed' argument
prefit_density	a named list of pre-fit 'densityMclust' objects for each sample, default is NULL
return_density	return the GMM parameter list or not (if applicable), default is FALSE

Value

A matrix containing the pairwise divergence or distance between all pairs of samples

Examples

```
# Bring in small example data of single cell embeddings
data(example_SCE_small)
sample_ids <- SingleCellExperiment::colData(example_SCE_small)$sample_id
pca_embeddings <- SingleCellExperiment::reducedDim(example_SCE_small,"PCA")
# Run gloscope on first 10 PCA embeddings
# We use 'KNN' option for speed ('GMM' is slightly slower)
pca_embeddings_subset <- pca_embeddings[,seq_len(10)] # select the first 10 PCs
dist_result <- gloscope(pca_embeddings_subset, sample_ids,
         dens="KNN", BPPARAM = BiocParallel::SerialParam(RNGseed=2))
dist_result
```

---

gloscopeProp *Calculate divergence between all sample pairs' cell type proportion*

Description

This function calculates a matrix of pairwise divergences between input samples' cell type proportion.

Usage

```
gloscopeProp(
  cell_sample_ids,
  cell_type_ids,
  ep = 0,
  dist_metric = c("KL", "JS", "TV")
)
```

Arguments

cell_sample_ids	a vector of the samples IDs each cell comes from. Length must match the number of element in ‘cell_type_ids‘
cell_type_ids	a vector of user defined cell type
ep	an numeric value added to the summary counts. Default ep = 0 means nothing will be added.
dist_metric	metric to calculate the divergence between samples.

Details

Options for ‘dist_metric‘ are as follows: "KL" calculates the symmetric-KL divergence. "JS" calculates the Jenson-Shannon distance. "TV" calculates the total variation distance.

Value

clusprop_dist a symmetric matrix of divergences

Examples

```
# Bring in small example data of single cell embeddings
data(example_SCE_small)
sample_id <- SingleCellExperiment::colData(example_SCE_small)$sample_id
cluster_id <- SingleCellExperiment::colData(example_SCE_small)$cluster_id
dist_result <- gloscopeProp(sample_id, cluster_id, ep = 0.5,
                                    dist_metric = "KL")
dist_result
```

---

plotCI *Plot confidence intervals*

Description

This function creates a ‘ggplot‘ object that plots the confidence intervals created by ‘bootCI‘

Usage

```
plotCI(ci_df, color_by, group_by, dodge_width = 0.5)
```

Arguments

ci_df	A data frame contains each sample’s metadata. Note this is NOT at the cell-level.
color_by	The column name or index in ci_df that should be used to color the confidence intervals by.

group_by	The column name or index in ci_df that should be used to determine how to group the confidence intervals. If missing all confidence intervals will be plotted in an order determined internally.
dodge_width	value passed to ‘width‘ argument of position_dodge if ‘group_by‘ variable is given. Controls separation between confidence intervals with the same grouping value.

Value

A plot of sample-pair divergences with confidence intervals

Examples

```
data(example_SCE_small)
sample_ids <- SingleCellExperiment::colData(example_SCE_small)$sample_id
# Run gloscope on first 10 PCA embeddings
# We use 'KNN' option for speed ('GMM' is slightly slower)
pca_embeddings <- SingleCellExperiment::reducedDim(example_SCE_small,"PCA")
pca_embeddings_subset <- pca_embeddings[,seq_len(10)] # select the first 10 PCs
dist_result <- gloscope(pca_embeddings_subset, sample_ids,
   dens="KNN",
   BPPARAM = BiocParallel::SerialParam(RNGseed=2))
# make a per-sample metadata
sample_metadata <- as.data.frame(unique(SingleCellExperiment::colData(example_SCE_small)[,c(1,2)]))
# make another variable
sample_metadata$fakeGroup<-c(rep(c("A","B"),each=2),"A")
manyboot<-bootCI(dist_result,
  sample_metadata,"sample_id",
  metrics=c("anosim","silhouette"),group_vars=c("phenotype","fakeGroup"),R=20)
plotCI(manyboot,group_by="metric",color_by="grouping")
```

---

plotHeatmap *Plot a heatmap of the GloScope represenation*

Description

This function creates a heatmap of the given GloScope divergence matrix.

Usage

```
plotHeatmap(
  dist_mat,
  metadata_df,
  sample_id,
  color_by,
  which_side = c("columns", "rows", "both"),
  ...
)
```

Arguments

dist_mat	The divergence matrix output of ‘gloscope()‘. Should be a symmetric, square matrix.
metadata_df	A data frame contains each sample’s metadata. Note this is NOT at the cell-level, and should have the same number of rows as dist_mat.
sample_id	The column name or index in metadata_df that contains the sample ID. This is for ensuring alignment between the dist_mat and the metadata_df. The row-names of dist_mat are expected to match the sample_id values.
color_by	A vector of column names or indices in metadata_df that should be used to color/annotate the samples.
which_side	One of "columns","rows", or "both", indicating whether the annotation of the samples in ‘color_by‘ should be on the rows, columns, or on both.
...	parameters passed to pheatmap.

Details

The function is a wrapper function to pheatmap. ‘color_by‘ is used to create subset of the ‘metadata_df‘ to pass to ‘annotation_col‘ (if ‘which_side="columns"‘) or ‘annotation_row‘ (if ‘which_side="rows"‘). If ‘which_side="both"‘, then it is passed to both, and ‘annotation_names_row‘ argument is set to ‘FALSE‘, suppressing labeling both the columns and rows (which user can thus not override). All other arguments to pheatmap can be passed directly by the user

Value

Invisibly returns the output of pheatmap

See Also

pheatmap

Examples

```
data(example_SCE_small)
sample_ids <- SingleCellExperiment::colData(example_SCE_small)$sample_id
# Run gloscope on first 10 PCA embeddings
# We use 'KNN' option for speed ('GMM' is slightly slower)
pca_embeddings <- SingleCellExperiment::reducedDim(example_SCE_small,"PCA")
pca_embeddings_subset <- pca_embeddings[,seq_len(10)] # select the first 10 PCs
dist_result <- gloscope(pca_embeddings_subset, sample_ids,
   dens="KNN",
   BPPARAM = BiocParallel::SerialParam(RNGseed=2))
# make a per-sample metadata
sample_metadata <- as.data.frame(unique(SingleCellExperiment::colData(example_SCE_small)[,c(1,2)]))
plotHeatmap(dist_mat = dist_result, metadata_df = sample_metadata ,
sample_id="sample_id", color_by="phenotype")
# Pass additional options to pheatmap to control colors of groups
library(RColorBrewer)
plotHeatmap(dist_mat = dist_result, metadata_df = sample_metadata ,
sample_id="sample_id", color_by="phenotype", which_side="both",
```

```
annotation_colors=list(phenotype = c(Covid = "magenta", Healthy = "white")),
color = colorRampPalette(brewer.pal(9, "PuBuGn"))(100))
```

`plotMDS` *Plot the multidimensional scaling of the GloScope represenation*

Description

This function calculates the multidimensional scaling for a GloScope divergence matrix and returns a ggplot object that plots it.

Usage

```
plotMDS(dist_mat, metadata_df, sample_id, k = 10, color_by, shape_by)
```

Arguments

`dist_mat`	The divergence matrix output of 'gloscope()'. Should be a symmetric, square matrix.
`metadata_df`	A data frame contains each sample's metadata. Note this is NOT at the cell-level, and should have the same number of rows as dist_mat.
`sample_id`	The column name or index in metadata_df that contains the sample ID. This is for ensuring alignment between the dist_mat and the metadata_df. The row-names of dist_mat are expected to match the sample_id values.
`k`	Number of MDS dimension to generate, default = 10
`color_by`	The column name or index in metadata_df that should be used to color the points by. If missing all points will be the same color.
`shape_by`	The column name or index in metadata_df that should be used to determine the shape of the points. If missing all points will be the same shape.

Details

The function calls `isoMDS` from the MASS package, calculates the requested k coordinates of the MDS plot. It also creates a ggplot object that will plot the first two dimensions color or shape coded by the given variables in the metadata data frame.

Value

A list containing the MDS embedding and plot of the distance matrix

- mds - A data.frame containing the MDS embedding, with the number of rows equal to the number of samples.
- plot - A ggplot object containing the plot object. 'print' of the object will create a plot.

See Also

`isoMDS`

Examples

```
data(example_SCE_small)
sample_ids <- SingleCellExperiment::colData(example_SCE_small)$sample_id
# Run gloscope on first 10 PCA embeddings
# We use 'KNN' option for speed ('GMM' is slightly slower)
pca_embeddings <- SingleCellExperiment::reducedDim(example_SCE_small,"PCA")
pca_embeddings_subset <- pca_embeddings[,seq_len(10)] # select the first 10 PCs
dist_result <- gloscope(pca_embeddings_subset, sample_ids,
   dens="KNN",
   BPPARAM = BiocParallel::SerialParam(RNGseed=2))
# make a per-sample metadata
sample_metadata <- as.data.frame(unique(SingleCellExperiment::colData(example_SCE_small)[,c(1,2)]))
mds_result <- plotMDS(dist_mat = dist_result, metadata_df = sample_metadata ,
sample_id="sample_id", color_by="phenotype",k=2)
head(mds_result$mds)
require(ggplot2)
mds_result$plot
# Add additional ggplot2 components to adapt figure
mds_result$plot + theme_bw()  + scale_color_manual(values=alpha(c("red","blue"),0.5))
```

Index

adonis2, *4, 5*
anosim, *5*

boot, *4, 5*
boot.ci, *4, 5*
bootCI *(*getMetrics*)*, 3
bootGloscope *(*getMetrics*)*, 3

example_SCE, 2
example_SCE_small *(*example_SCE*)*, 2

getMetrics, 3
gloscope, 6
gloscopeProp, 7

how, *4, 5*

isoMDS, *11*

pheatmap, *10*
plotCI, 8
plotHeatmap, 9
plotMDS, 11
position_dodge, *9*

silhouette, *5*