Package 'ChromSCape'

October 30, 2025

Title Analysis of single-cell epigenomics datasets with a Shiny App

Version 1.21.0

Description ChromSCape - Chromatin landscape profiling for Single Cells - is a ready-to-launch user-friendly Shiny Application for the analysis of single-cell epigenomics datasets (scChIP-seq, scATAC-seq, scCUT&Tag, ...) from aligned data to differential analysis & gene set enrichment analysis. It is highly interactive, enables users to save their analysis and covers a wide range of analytical steps: QC, preprocessing, filtering, batch correction, dimensionality reduction, vizualisation, clustering, differential analysis and gene set analysis.

License GPL-3

biocViews ShinyApps, Software, SingleCell, ChIPSeq, ATACSeq, MethylSeq, Classification, Clustering, Epigenetics, PrincipalComponent, SingleCell, ATACSeq, ChIPSeq, Annotation, BatchEffect, MultipleComparison, Normalization, Pathways, Preprocessing, QualityControl, ReportWriting, Visualization, GeneSetEnrichment, DifferentialPeakCalling

VignetteBuilder knitr

URL https://github.com/vallotlab/ChromSCape

BugReports https://github.com/vallotlab/ChromSCape/issues

Encoding UTF-8

LazyData false

Suggests testthat, knitr, markdown, rmarkdown, BiocStyle, Signac, future, igraph, bluster, httr

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

Imports shiny, colourpicker, shinyjs, rtracklayer, shinyFiles, shinyhelper, shinyWidgets, shinydashboardPlus, shinycssloaders, Matrix, plotly, shinydashboard, colorRamps, kableExtra, viridis, batchelor, BiocParallel, parallel, Rsamtools, ggplot2, ggrepel, gggenes, gridExtra, qualV, stringdist, stringr, fs, qs, DT, scran, scater, ConsensusClusterPlus, Rtsne, dplyr, tidyr, GenomicRanges, IRanges, irlba, rlist, umap, tibble,

methods, jsonlite, edgeR, stats, graphics, grDevices, utils, S4Vectors, SingleCellExperiment, SummarizedExperiment, msigdbr, forcats, Rcpp, coop, matrixTests, DelayedArray

Depends R (>= 4.1)

LinkingTo Rcpp

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

git_branch devel

git_last_commit 47d0ff7

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2025-10-30

Author Pacome Prompsy [aut, cre] (ORCID: <https://orcid.org/0000-0003-4375-7583>), Celine Vallot [aut] (ORCID: <https://orcid.org/0000-0003-1601-2359>)

Maintainer Pacome Prompsy <pacome.pr@gmail.com>

Contents

`annotation_from_merged_peaks` *Find nearest peaks of each gene and return refined annotation*

Description

Find nearest peaks of each gene and return refined annotation

Usage

```
annotation_from_merged_peaks(scExp, odir, merged_peaks, geneTSS_annotation)
```

Arguments

`scExp`	A SingleCellExperiment object
`odir`	An output directory where to write the mergedpeaks BED file
`merged_peaks`	A list of GRanges object containing the merged peaks
`geneTSS_annotation`	A GRanges object with reference genes

Value

A data.frame with refined annotation

annotToCol2 *annotToCol2*

Description

annotToCol2

Usage

```
annotToCol2(
  annotS = NULL,
  annotT = NULL,
  missing = c("", NA),
  anotype = NULL,
  maxnumcateg = 2,
  categCol = NULL,
  quantitCol = NULL,
  plotLegend = TRUE,
  plotLegendFile = NULL
)
```

Arguments

annotS	A color matrix
annotT	A color matrix
missing	Convert missing to NA
anotype	Annotation type
maxnumcateg	Maximum number of categories
categCol	Categorical columns
quantitCol	Quantitative columns
plotLegend	Plot legend ?
plotLegendFile	Which file to plot legend ?

Value

A matrix of continuous or discrete colors

Examples

```
data("scExp")
annotToCol2(SingleCellExperiment::colData(scExp), plotLegend = FALSE)
```

anocol_binary *Helper binary column for anocol function*

Description

Helper binary column for anocol function

Usage

```
anocol_binary(anocol, anotype, plotLegend, annotS)
```

Arguments

anocol	The color feature matrix
anotype	The feature types
plotLegend	Plot legend ?
annotS	A color matrix

Value

A color matrix similar to anocol with binrary columns colored

anocol_categorical *Helper binary column for anocol function*

Description

Helper binary column for anocol function

Usage

```
anocol_categorical(anocol, categCol, anotype, plotLegend, annotS)
```

Arguments

anocol	The color feature matrix
categCol	Colors for categorical features
anotype	The feature types
plotLegend	Plot legend ?
annotS	A color matrix

Value

A color matrix similar to anocol with binrary columns colored

---

`bams_to_matrix_indexes`

Count bam files on interval to create count indexes

Description

Count bam files on interval to create count indexes

Usage

```
bams_to_matrix_indexes(dir, which, BPPARAM = BiocParallel::bpparam())
```

Arguments

`dir`	A directory containing single cell BAM files and BAI files
`which`	Genomic Range on which to count
`BPPARAM`	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A list containing a "feature index" data.frame and a count vector for non 0 entries, both used to form the sparse matrix

---

`beds_to_matrix_indexes`

Count bed files on interval to create count indexes

Description

Count bed files on interval to create count indexes

Usage

```
beds_to_matrix_indexes(dir, which, BPPARAM = BiocParallel::bpparam())
```

Arguments

`dir`	A directory containing the single cell BED files
`which`	Genomic Range on which to count
`BPPARAM`	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A list containing a "feature index" data.frame and a names of cells as vector both used to form the sparse matrix

`calculate_CNA` *Estimate copy number alterations in cytobands*

Description

Cytobands are considered large enough in order that a variation at the cytoband level is not considered as an epigenetic event but as a genetic event, e.g. Copy Number Alterations. The function successively :

- Calculates the fraction of reads in each cytoband (FrCyto). See `calculate_cyto_mat`
- Calculates the log2-ratio FrCyto of each cell by the average FrCyto in normal cells. See `calculate_logRatio_CNA`
- Estimates if there was a gain or a loss of copy in each cyto band. See `calculate_gain_or_loss`

The corresponding matrices are accessibles in the reducedDim slots "cytoBands", "logRatio_cytoBands" and "gainOrLoss_cytoBands" respectively.

Usage

```
calculate_CNA(
  scExp,
  control_samples = unique(scExp$sample_id)[1],
  ref_genome = c("hg38", "mm10", "ce11")[1],
  quantiles_to_define_gol = c(0.05, 0.95)
)
```

Arguments

`scExp`	A SingleCellExperiment with "logRatio_cytoBand" reducedDim slot filled. See `calculate_logRatio_CNA`
`control_samples`	Sample IDs of the normal sample to take as reference.
`ref_genome`	Reference genome ('hg38' or 'mm10')
`quantiles_to_define_gol`	Quantiles of normal log2-ratio distribution below/above which cytoband is considered to be a loss/gain. (c(0.05,0.95)). See `calculate_gain_or_loss`

Value

The SCE with the fraction of reads, log2-ratio and gain or loss in each cytobands in each cells (of dimension cell x cytoband) in the reducedDim slots.

Examples

```
data("scExp")
scExp = calculate_CNA(scExp,  control_samples = unique(scExp$sample_id)[1],
ref_genome="hg38", quantiles_to_define_gol = c(0.05,0.95))
SingleCellExperiment::reducedDim(scExp, "cytoBand")
SingleCellExperiment::reducedDim(scExp, "logRatio_cytoBand")
SingleCellExperiment::reducedDim(scExp, "gainOrLoss_cytoBand")
```

---

calculate_cyto_mat *Calculate Fraction of reads in each cytobands*

Description

Re-Count binned reads onto cytobands and calculate the fraction of reads in each of the cytoband in each cell. For each cell, the fraction of reads in any given cytoband is calculated. Cytobands are considered large enough in order that a variation at the cytoband level is not considered as an epigenetic event but as a genetic event, e.g. Copy Number Alterations.

Usage

```
calculate_cyto_mat(scExp, ref_genome = c("hg38", "mm10", "ce11")[1])
```

Arguments

scExp	A SingleCellExperiment with genomic coordinate as features (peaks or bins)
ref_genome	Reference genome ('hg38' or 'mm10')

Value

The SCE with the fraction of reads in each cytobands in each cells (of dimension cell x cytoband) in the reducedDim slot "cytoBand".

Examples

```
data("scExp")
scExp = calculate_cyto_mat(scExp, ref_genome="hg38")
SingleCellExperiment::reducedDim(scExp, "cytoBand")
```

`calculate_gain_or_loss` *Estimate the copy gains/loss of tumor vs normal based on log2-ratio of fraction of reads*

Description

Given a SingleCellExperiment object with the slot "logRatio_cytoBand" containing the log2-ratio of the fraction of reads in each cytoband, estimate if the cytoband was lost or acquired a gain in a non-quantitative way. To do so, the quantiles distribution of the normal cells are calculated, and any cytoband below or above will be considered as a loss/gain. The False Discovery Rate is directly proportional to the quantiles.

Usage

```
calculate_gain_or_loss(scExp, controls, quantiles = c(0.05, 0.95))
```

Arguments

`scExp`	A SingleCellExperiment with "logRatio_cytoBand" reducedDim slot filled. See `calculate_logRatio_CNA`
`controls`	Sample IDs or Cell IDs of the normal sample to take as reference.
`quantiles`	Quantiles of normal log2-ratio distribution below/above which cytoband is considered to be a loss/gain. (c(0.05,0.95))

Value

The SCE with the gain or loss in each cytobands in each cells (of dimension cell x cytoband) in the reducedDim slot "gainOrLoss_cytoBand".

Examples

```
data("scExp")
scExp = calculate_cyto_mat(scExp, ref_genome="hg38")
scExp = calculate_logRatio_CNA(scExp, controls=unique(scExp$sample_id)[1])
scExp = calculate_gain_or_loss(scExp, controls=unique(scExp$sample_id)[1])
SingleCellExperiment::reducedDim(scExp, "gainOrLoss_cytoBand")
```

`calculate_logRatio_CNA` *Calculate the log2-ratio of tumor vs normal fraction of reads in cytobands*

Description

Given a SingleCellExperiment object with the slot "cytoBand" containing the fraction of reads in each cytoband, calculates the log2-ratio of tumor vs normal fraction of reads in cytobands, cell by cell. If the average signal in normal sample in a cytoband is 0, set this value to 1 so that the ratio won't affect the fraction of read value.

Usage

```
calculate_logRatio_CNA(scExp, controls)
```

Arguments

`scExp`	A SingleCellExperiment with "cytoBand" reducedDim slot filled. • see `calculate_cyto_mat`
`controls`	Sample IDs or Cell IDs of the normal sample to take as reference.

Value

The SCE with the log2-ratio of fraction of reads in each cytobands in each cells (of dimension cell x cytoband) in the reducedDim slot "logRatio_cytoBand".

Examples

```
data("scExp")
scExp = calculate_cyto_mat(scExp, ref_genome="hg38")
scExp = calculate_logRatio_CNA(scExp, controls=unique(scExp$sample_id)[1])
SingleCellExperiment::reducedDim(scExp, "logRatio_cytoBand")
```

`call_macs2_merge_peaks` *Calling MACS2 peak caller and merging resulting peaks*

Description

Calling MACS2 peak caller and merging resulting peaks

Usage

```
call_macs2_merge_peaks(
  affectation,
  odir,
  p.value,
  format = c("scBED", "BAM")[1],
  ref,
  peak_distance_to_merge
)
```

Arguments

affectation	Annotation data.frame with cell cluster and cell id information
odir	Output directory to write MACS2 output
p.value	P value to detect peaks, passed to MACS2
format	File format, either "BAM" or "scBED"
ref	Reference genome to get chromosome information from.
peak_distance_to_merge	Distance to merge peaks

Value

A list of merged GRanges peaks

---

ce11.chromosomes *Data.frame of chromosome length - ce11*

Description

This data frame provides the length of each "canonical" chromosomes of C Elegans genome build ce11.

Usage

```
data("ce11.chromosomes")
```

Format

ce11.chromosomes - a data frame with 7 chromosomes and 3 variables:

chr Chromosome - character

start Start of the chromosome (bp) - integer

end End of the chromosome (bp) - integer

Value

ce11.chromosomes - a data frame with 7 chromosomes and 3 variables

ce11.GeneTSS *Data.frame of gene TSS - ce11*

Description

This dataframe was extracted from Gencode v25 and report the Transcription Start Site of each gene in the C Elegans genome build ce11

Usage

```
data("ce11.GeneTSS")
```

Format

ce11.GeneTSS - a data frame with 20,051 genes and 5 variables:

chr Chromosome - character

start Start of the gene (TSS) - integer

end End of the gene - integer

Gene Gene symbol - character

strand Srand - character

Value

ce11.GeneTSS - a data frame with 20,051 genes and 5 variables

changeRange *changeRange*

Description

changeRange

Usage

```
changeRange(v, newmin = 1, newmax = 10)
```

Arguments

v	A numeric vector
newmin	New min
newmax	New max

Value

A matrix with values scaled between newmin and newmax

CheA3_TF_nTargets *A data.frame with the number of targets of each TF in ChEA3*

Description

This data.frame was obtained by downloading datasets from ChEA3 database (https://maayanlab.cloud/chea3/) and merging targets for :

- ARCHS4_Coexpression
- ENCODE_ChIP-seq
- Enrichr_Queries
- GTEx_Coexpression
- Literature_ChIP-seq
- ReMap_ChIP-seq

Usage

```
data("CheA3_TF_nTargets")
```

Format

CheA3_TF_nTargets - a data.frame with 1632 rows (unique TFs) and 2 columns

Value

A CheA3_TF_nTargets - a data.frame with 1632 rows (unique TFs) and 2 columns

References

Keenan AB, Torre D, Lachmann A, Leong AK, Wojciechowicz M, Utti V, Jagodnik K, Kropiwnicki E, Wang Z, Ma'ayan A (2019) ChEA3: transcription factor enrichment analysis by orthogonal omics integration. Nucleic Acids Research. doi: 10.1093/nar/gkz446

The data.frame is composed of two columns:

- TF column containing the TF gene names (human)
- nTargets_TF containing the number of targets for this TF in the combined database.

Examples

```
data("CheA3_TF_nTargets")
head(CheA3_TF_nTargets)
```

`check_correct_datamatrix` *Check if matrix rownames are well formated and correct if needed*

Description

Throws warnings / error if matrix is in the wrong format

Usage

```
check_correct_datamatrix(datamatrix_single, sample_name = "")
```

Arguments

`datamatrix_single`	A sparse matrix
`sample_name`	Matrix sample name for warnings

Value

A sparseMatrix in the right rownames format

`choose_cluster_scExp` *Choose a number of clusters*

Description

This functions takes as input a SingleCellExperiment object and a number of cluster to select. It outputs a SingleCellExperiment object with each cell assigned to a correlation cluster in colData. Also calculates a hierarchical clustering of the consensus associations calculated by ConsensusClusterPlus.

Usage

```
choose_cluster_scExp(
  scExp,
  nclust = 3,
  consensus = FALSE,
  hc_linkage = "ward.D"
)
```

Arguments

`scExp`	A SingleCellExperiment object containing consclust in metadata.
`nclust`	Number of cluster to pick (3)
`consensus`	Use consensus clustering results instead of simple hierarchical clustering ? (FALSE)
`hc_linkage`	A linkage method for hierarchical clustering. See cor. ('ward.D')

Value

Returns a SingleCellExperiment object with each cell assigned to a correlation cluster in colData.

Examples

```
data("scExp")
scExp_cf = correlation_and_hierarchical_clust_scExp(scExp)
scExp_cf = choose_cluster_scExp(scExp_cf,nclust=3,consensus=FALSE)
table(scExp_cf$cell_cluster)

scExp_cf = consensus_clustering_scExp(scExp)
scExp_cf_consensus = choose_cluster_scExp(scExp_cf,nclust=3,consensus=TRUE)
table(scExp_cf_consensus$cell_cluster)
```

---

choose_perplexity *Choose perplexity depending on number of cells for Tsne*

Description

Choose perplexity depending on number of cells for Tsne

Usage

```
choose_perplexity(dataset)
```

Arguments

dataset	A matrix of features x cells (rows x columns)

Value

A number between 5 and 30 to use in Rtsne function

---

col2hex *Col2Hex*

Description

Transform character color to hexadecimal color code.

Usage

```
col2hex(cname)
```

Arguments

cname	Color name

Value

The HEX color code of a particular color

---

colors_scExp *Adding colors to cells & features*

Description

Adding colors to cells & features

Usage

```
colors_scExp(
  scExp,
  annotCol = "sample_id",
  color_by = "sample_id",
  color_df = NULL
)
```

Arguments

scExp	A SingleCellExperiment Object
annotCol	Column names to color
color_by	If specifying color_df, column names to color
color_df	Color data.frame to specify which color for which condition

Value

A SingleCellExperiment with additionnal "color" columns in colData

Examples

```
data("scExp")
scExp = colors_scExp(scExp,annotCol = c("sample_id",
"total_counts"),
 color_by =  c("sample_id","total_counts"))

#Specific colors using a manually created data.frame :
color_df = data.frame(sample_id=unique(scExp$sample_id),
 sample_id_color=c("red","blue","green","yellow"))
scExp = colors_scExp(scExp,annotCol="sample_id",
color_by="sample_id",color_df=color_df)
```

combine_datamatrix *Combine two matrices and emit warning if no regions are in common*

Description

Combine two matrices and emit warning if no regions are in common

Usage

```
combine_datamatrix(datamatrix, datamatrix_single, file_names, i)
```

Arguments

datamatrix	A sparse matrix or NULL if empty
datamatrix_single	Another sparse matrix
file_names	File name corresponding to the matrix for warnings
i	file number

Value

A combined sparse matrix

combine_enrichmentTests *Run enrichment tests and combine into list*

Description

Run enrichment tests and combine into list

Usage

```
combine_enrichmentTests(
  diff,
  enrichment_qval,
  qval.th,
  logFC.th,
  min.percent,
  annotFeat_long,
  peak_distance,
  refined_annotation,
  GeneSets,
  GeneSetsDf,
  GenePool,
  progress = NULL
)
```

Arguments

diff	Differential list
enrichment_qval	Adusted p-value threshold above which a pathway is considered significative list
qval.th	Differential analysis adjusted p.value threshold
logFC.th	Differential analysis log-fold change threshold
min.percent	Minimum fraction of cells having the feature active to consider it as significantly differential. (0.01)
annotFeat_long	Long annotation
peak_distance	Maximum gene to peak distance
refined_annotation	Refined annotation data.frame if peak calling is done
GeneSets	List of pathways
GeneSetsDf	Data.frame of pathways
GenePool	Pool of possible genes for testing
progress	A shiny Progress instance to display progress bar.

Value

A list of list of pathway enrichment data.frames for Both / Over / Under and for each cluster

---

comparable_variables *Find comparable variable scExp*

Description

Find comparable variable scExp

Usage

```
comparable_variables(scExp, allExp = TRUE)
```

Arguments

scExp	A SingleCellExperiment
allExp	A logical indicating wether alternative experiments comparable variables should also be fetch.

Value

A character vector with the comparable variable names

CompareedgeRGLM *Creates a summary table with the number of genes under- or overexpressed in each group and outputs several graphical representations*

Description

Creates a summary table with the number of genes under- or overexpressed in each group and outputs several graphical representations

Usage

```
CompareedgeRGLM(
  dataMat = NULL,
  annot = NULL,
  ref_group = NULL,
  groups = NULL,
  featureTab = NULL,
  norm_method = "TMMwsp"
)
```

Arguments

dataMat	reads matrix
annot	selected annotation of interest
ref_group	List containing one or more vectors of reference samples. Name of the vectors will be used in the results table. The length of this list should be 1 or the same length as the groups list
groups	List containing the IDs of groups to be compared with the reference samples. Names of the vectors will be used in the results table
featureTab	Feature annotations to be added to the results table
norm_method	Which method to use for normalizing ('upperquantile')

Value

A dataframe containing the foldchange and p.value of each feature

Author(s)

Eric Letouze & Celine Vallot

Examples

```
data("scExp")
scExp_cf = correlation_and_hierarchical_clust_scExp(scExp)
scExp_cf = choose_cluster_scExp(scExp_cf,nclust=2,consensus=FALSE)
featureTab = as.data.frame(SummarizedExperiment::rowRanges(scExp_cf))
```

```
rownames(featureTab) = featureTab$ID
ref_group = list("C1"=scExp_cf$cell_id[which(scExp_cf$cell_cluster=="C1")])
groups = list("C2"=scExp_cf$cell_id[which(scExp_cf$cell_cluster=="C2")])
myres = CompareedgeRGLM(as.matrix(SingleCellExperiment::counts(scExp_cf)),
annot=as.data.frame(SingleCellExperiment::colData(scExp_cf)),
   ref_group=ref_group,groups=groups, featureTab=featureTab)
```

CompareWilcox *CompareWilcox*

Description

CompareWilcox

Usage

```
CompareWilcox(
  dataMat = NULL,
  annot = NULL,
  ref_group = NULL,
  groups = NULL,
  featureTab = NULL,
  block = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

dataMat	A raw count matrix
annot	A cell annotation data.frame
ref_group	List with cells in reference group(s)
groups	List with cells in group(s) to test
featureTab	data.frame with feature annotation
block	Use a blocking factor to conteract batch effect ?
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A dataframe containing the foldchange and p.value of each feature

Author(s)

Eric Letouze & Celine Vallot & Pacome Prompsy

Examples

```
data("scExp")
scExp_cf = correlation_and_hierarchical_clust_scExp(scExp)
scExp_cf = choose_cluster_scExp(scExp_cf,nclust=2,consensus=FALSE)
featureTab = as.data.frame(SummarizedExperiment::rowRanges(scExp_cf))
rownames(featureTab) = featureTab$ID
ref_group = list("C1"=scExp_cf$cell_id[which(scExp_cf$cell_cluster=="C1")])
groups = list("C2"=scExp_cf$cell_id[which(scExp_cf$cell_cluster=="C2")])
myres = CompareWilcox(as.matrix(SingleCellExperiment::normcounts(scExp_cf)),
annot=as.data.frame(SingleCellExperiment::colData(scExp_cf)),
   ref_group=ref_group,groups=groups, featureTab=featureTab)
```

---

concatenate_scBed_into_clusters

Concatenate single-cell BED into clusters

Description

Concatenate single-cell BED into clusters

Usage

```
concatenate_scBed_into_clusters(affectation, files_list, odir)
```

Arguments

affectation	Annotation data.frame containing cluster information
files_list	Named list of scBED file paths to concatenate. List Names must match affectation$sample_id and basenames must match affectation$barcode.
odir	Output directory to write concatenate pseudo-bulk BEDs.

Value

Merge single-cell BED files into cluster BED files. Ungzip file if BED is gzipped.

`consensus_clustering_scExp` *Wrapper to apply ConsensusClusterPlus to scExp object*

Description

Runs consensus hierarchical clustering on PCA feature space of scExp object. Plot consensus scores for each number of clusters. See ConsensusClusterPlus - Wilkerson, M.D., Hayes, D.N. (2010). ConsensusClusterPlus: a class discovery tool with confidence assessments and item tracking. Bioinformatics, 2010 Jun 15;26(12):1572-3.

Usage

```
consensus_clustering_scExp(
  scExp,
  prefix = NULL,
  maxK = 10,
  reps = 100,
  pItem = 0.8,
  pFeature = 1,
  distance = "pearson",
  clusterAlg = "hc",
  innerLinkage = "ward.D",
  finalLinkage = "ward.D",
  plot_consclust = "pdf",
  plot_icl = "png"
)
```

Arguments

`scExp`	A SingleCellExperiment object containing 'PCA' in reducedDims.
`prefix`	character value for output directory. Directory is created only if plot_consclust is not NULL. This title can be an abosulte or relative path.
`maxK`	integer value. maximum cluster number to evaluate. (10)
`reps`	integer value. number of subsamples. (100)
`pItem`	numerical value. proportion of items to sample. (0.8)
`pFeature`	numerical value. proportion of features to sample. (1)
`distance`	character value. 'pearson': (1 - Pearson correlation), 'spearman' (1 - Spearman correlation), 'euclidean', 'binary', 'maximum', 'canberra', 'minkowski' or custom distance function. ('pearson')
`clusterAlg`	character value. cluster algorithm. 'hc' heirarchical (hclust), 'pam' for paritioning around medoids, 'km' for k-means upon data matrix, 'kmdist' ('hc') for k-means upon distance matrices (former km option), or a function that returns a clustering. ('hc')

innerLinkage	hierarchical linkage method for subsampling. ('ward.D')
finalLinkage	hierarchical linkage method for consensus matrix. ('ward.D')
plot_consclust	character value. NULL - print to screen, 'pdf', 'png', 'pngBMP' for bitmap png, helpful for large datasets. ('pdf')
plot_icl	same as above for item consensus plot. ('png')

Details

This functions takes as input a SingleCellExperiment object that must have 'PCA' in reducedDims and outputs a SingleCellExperiment object containing consclust list calculated cluster consensus and item consensus scores in metadata.

Value

Returns a SingleCellExperiment object containing consclust list, calculated cluster consensus and item consensus scores in metadata.

References

ConsensusClusterPlus package by Wilkerson, M.D., Hayes, D.N. (2010). ConsensusClusterPlus: a class discovery tool with confidence assessments and item tracking. Bioinformatics, 2010 Jun 15;26(12):1572-3.

Examples

```
data("scExp")
scExp_cf = correlation_and_hierarchical_clust_scExp(scExp)
scExp_cf = consensus_clustering_scExp(scExp)
```

---

correlation_and_hierarchical_clust_scExp

Correlation and hierarchical clustering

Description

Calculates cell to cell correlation matrix based on the PCA feature space and runs hierarchical clustering taking 1 - correlation scores as distance.

Usage

```
correlation_and_hierarchical_clust_scExp(scExp, hc_linkage = "ward.D")
```

Arguments

scExp	A SingleCellExperiment object, containing 'PCA' in reducedDims.
hc_linkage	A linkage method for hierarchical clustering. See cor. ('ward.D')

Details

This functions takes as input a SingleCellExperiment object that must have PCA calculated and outputs a SingleCellExperiment object with correlation matrix and hierarchical clustering.

Value

Return a SingleCellExperiment object with correlation matrix & hiearchical clustering.

Examples

```
data("scExp")
scExp_cf = correlation_and_hierarchical_clust_scExp(scExp)
```

---

count_coverage *Create a smoothed and normalized coverage track from a BAM file and given a bin GenomicRanges object (same as deepTools bamCoverage)*

Description

Normalization is CPM, smoothing is done by averaging on n_smoothBin regions left and right of any given region.

Usage

```
count_coverage(
  input,
  format = "BAM",
  bins,
  canonical_chr,
  norm_factor,
  n_smoothBin = 5,
  ref = "hg38",
  read_size = 101,
  original_bins = NULL
)
```

Arguments

input	Either a named list of character vector of path towards single-cell BED files or a sparse raw matrix of small bins («500bp). If a named list specifying scBEDn the names MUST correspond to the 'sample_id' column in your SingleCellExperiment object. The single-cell BED files names MUST match the barcode names in your SingleCellExperiment (column 'barcode'). The scBED files can be gzipped or not.
format	File format, either "BAM" or "BED"
bins	A GenomicRanges object of binned genome

canonical_chr	GenomicRanges of the chromosomes to read the BAM file.
norm_factor	Then number of cells or total number of reads in the given sample, for normalization.
n_smoothBin	Number of bins left and right to smooth the signal.
ref	Genomic reference
read_size	Length of the reads
original_bins	Original bins GenomicRanges in case the format is raw matrix.

Value

A binned GenomicRanges that can be readily exported into bigwig file.

`create_project_folder` *Create ChromSCape project folder*

Description

Creates a project folder that will be recognizable by ChromSCape Shiny application.

Usage

```
create_project_folder(
  output_directory,
  analysis_name = "Analysis_1",
  ref_genome = c("hg38", "mm10", "ce11")[1]
)
```

Arguments

output_directory	Path towards the directory to create the 'ChromSCape_Analyses' folder and the analysis subfolder. If this path already contains the 'ChromSCape_Analyses' folder, will only create the analysis subfolder.
analysis_name	Name of the analysis. Must only contain alphanumerical characters or '_'.
ref_genome	Reference genome, either 'hg38' or 'mm10'.

Value

Creates the project folder and returns the root of the project.

Examples

```
dir = tempdir()
create_project_folder(output_directory = dir,
 analysis_name = "Analysis_1")
list.dirs(file.path(dir))
```

create_sample_name_mat *Create a sample name matrix*

Description

Create a sample name matrix

Usage

```
create_sample_name_mat(nb_samples, samples_names)
```

Arguments

nb_samples	Number of samples
samples_names	Character vector of sample names

Value

A matrix

create_scDataset_raw *Create a simulated single cell datamatrix & cell annotation*

Description

Create a simulated single cell datamatrix & cell annotation

Usage

```
create_scDataset_raw(
  cells = 300,
  features = 600,
  featureType = c("window", "peak", "gene"),
  sparse = TRUE,
  nsamp = 4,
  ref = "hg38",
  batch_id = factor(rep(1, nsamp))
)
```

Arguments

cells	Number of cells (300)
features	Number of features (600)
featureType	Type of feature (window)
sparse	Is matrix sparse ? (TRUE)
nsamp	Number of samples (4)
ref	Reference genome ('hg38')
batch_id	Batch origin (factor((1,1,1,1))

Value

A list composed of * mat : a sparse matrix following an approximation of the negative binomial law (adapted to scChIPseq) * annot : a data.frame of cell annotation * batches : an integer vector with the batch number for each cell

Examples

```
# Creating a basic sparse 600 genomic bins x 300 cells matrix and annotation
l = create_scDataset_raw()
head(l$mat)
head(l$annot)
head(l$batches)

# Specifying number of cells, features and samples
l2 = create_scDataset_raw(cells = 500, features = 500, nsamp=2)

# Specifying species
mouse_l = create_scDataset_raw(ref="mm10")

# Specifying batches
batch_l = create_scDataset_raw(nsamp=4, batch_id = factor(c(1,1,2,2)))

# Peaks of different size as features
peak_l = create_scDataset_raw(featureType="peak")
head(peak_l$mat)

# Genes as features
gene_l = create_scDataset_raw(featureType="gene")
head(gene_l$mat)
```

---

create_scExp *Wrapper to create the single cell experiment from count matrix and feature dataframe*

Description

Create the single cell experiment from (sparse) datamatrix and feature dataframe containing feature names and location. Also optionally removes zero count Features, zero count Cells, non canconical chromosomes, and chromosome M. Calculates QC Metrics (scran).

Usage

```
create_scExp(
  datamatrix,
  annot,
  remove_zero_cells = TRUE,
  remove_zero_features = TRUE,
  remove_non_canonical = TRUE,
  remove_chr_M = TRUE,
  mainExpName = "main",
  verbose = TRUE
)
```

Arguments

datamatrix	A matrix or sparseMatrix of raw counts. Features x Cells (rows x columns).
annot	A data.frame containing informations on cells. Should have the same number of rows as the number of columns in datamatrix.
remove_zero_cells	remove cells with zero counts ? (TRUE)
remove_zero_features	remove cells with zero counts ? (TRUE)
remove_non_canonical	remove non canonical chromosomes ?(TRUE)
remove_chr_M	remove chromosomes M ? (TRUE)
mainExpName	Name of the mainExpName e.g. 'bins', 'peaks'... ("default")
verbose	(TRUE)

Value

Returns a SingleCellExperiment object.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp
```

DA_custom *Differential Analysis Custom in 'One vs One' mode*

Description

Differential Analysis Custom in 'One vs One' mode

Usage

```
DA_custom(
  affectation,
  by,
  counts,
  method,
  feature,
  block,
  ref,
  group,
  progress = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

affectation	An annotation data.frame with cell_id and
by	= A character specifying the column of the object containing the groups of cells to compare.
counts	Count matrix
method	DA method : Wilcoxon or EdgeR
feature	Feature tables
block	Blocking feature
ref	If de_type is custom, the reference to compare (data.frame), must be a one-column data.frame with cell_clusters or sample_id as character in rows
group	If de_type is custom, the group to compare (data.frame), must be a one-column data.frame with cell_clusters or sample_id as character in rows
progress	A shiny Progress instance to display progress bar.
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A list of results, groups compared and references

DA_one_vs_rest *Differential Analysis in 'One vs Rest' mode*

Description

Differential Analysis in 'One vs Rest' mode

Usage

```
DA_one_vs_rest(
  affectation,
  by,
  counts,
  method,
  feature,
  block,
  progress = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

affectation	An annotation data.frame with cell_id and cell_cluster columns
by	= A character specifying the column of the object containing the groups of cells to compare.
counts	Count matrix
method	DA method : Wilcoxon or EdgeR
feature	Feature tables
block	Blocking feature
progress	A shiny Progress instance to display progress bar.
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A list of results, groups compared and references

DA_pairwise *Run differential analysis in Pairwise mode*

Description

Run differential analysis in Pairwise mode

Usage

```
DA_pairwise(
  affectation,
  by,
  counts,
  method,
  feature,
  block,
  progress = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

affectation	An annotation data.frame with cell_cluster and cell_id columns
by	= A character specifying the column of the object containing the groups of cells to compare.
counts	Count matrix
method	DA method, Wilcoxon or edgeR
feature	Feature data.frame
block	Blocking feature
progress	A shiny Progress instance to display progress bar.
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A list of results, groups compared and references

define_feature *Define the features on which reads will be counted*

Description

Define the features on which reads will be counted

Usage

```
define_feature(ref = c("hg38","mm10", "ce11")[1],
 peak_file = NULL,
 bin_width  = NULL,
 genebody = FALSE,
 extendPromoter = 2500)
```

Arguments

ref	Reference genome
peak_file	A bed file if counting on peaks
bin_width	A number of bins if divinding genome into fixed width bins
genebody	A logical indicating if feature should be counted in genebodies and promoter.
extendPromoter	Extension length before TSS (2500).

Value

A GRanges object

Examples

```
gr_bins = define_feature("hg38", bin_width = 50000)
gr_genes = define_feature("hg38", genebody = TRUE, extendPromoter = 5000)
```

detect_samples *Heuristic discovery of samples based on cell labels*

Description

Identify a fixed number of common string (samples) in a set of varying strings (cells). E.g. in the set "Sample1_cell1","Sample1_cell2","Sample2_cell1","Sample2_cell2" and with nb_samples=2, the function returns "Sample1","Sample1","Sample2","Sample2".

Usage

```
detect_samples(barcodes, nb_samples = 1)
```

Arguments

barcodes	Vector of cell barcode names (e.g. Sample1_cell1, Sample1_cell2...)
nb_samples	Number of samples to find

Value

character vector of sample names the same length as cell labels

Examples

```
barcodes = c(paste0("HBCx22_BC_",seq_len(100)),
paste0("mouse_sample_XX",208:397))
samples = detect_samples(barcodes, nb_samples=2)
```

---

differential_activation

Find Differentialy Activated Features (One vs All)

Description

Based on the statement that single-cell epigenomic dataset are very sparse, specifically when analysis small bins or peaks, we can define each feature as being 'active' or not simply by the presence or the absence of reads in this feature. This is the equivalent of binarize the data. When trying to find differences in signal for a feature between multiple cell groups, this function simply compare the percentage of cells 'activating' the feature in each of the group. The p.values are then calculated using a Pearson's Chi-squared Test for Count Data (comparing the number of active cells in one group vs the other) and corrected using Benjamini-Hochberg correction for multiple testing.

Usage

```
differential_activation(
  scExp,
  by = c("cell_cluster", "sample_id")[1],
  verbose = TRUE,
  progress = NULL
)
```

Arguments

scExp	A SingleCellExperiment object containing consclust with selected number of cluster.
by	Which grouping to run the marker enrichment ?
verbose	Print ?
progress	A shiny Progress instance to display progress bar.

Details

To calculate the logFC, the percentage of activation of the features are corrected for total number of reads to correct for library size bias. For each cluster ('group') the function consider the rest of the cells as the reference.

Value

Returns a dataframe of differential activation results that contains the rowData of the SingleCellExperiment with additional logFC, q.value, group activation (fraction of cells active for each feature in the group cells), reference activation (fraction of cells active for each feature in the reference cells).

See Also

For Pearson's Chi-squared Test for Count Data chisq.test. For other differential analysis see differential_analysis_scExp.

Examples

```
data("scExp")
res = differential_activation(scExp, by = "cell_cluster")
res = differential_activation(scExp, by = "sample_id")
```

---

differential_analysis_scExp

Runs differential analysis between cell clusters

Description

Based on clusters of cell defined previously, runs non-parametric Wilcoxon Rank Sum test to find significantly depleted or enriched features, in 'one_vs_rest' mode or 'pairwise' mode. In pairwise mode, each cluster is compared to all other cluster individually, and then pairwise comparisons between clusters are combined to find overall differential features using combineMarkers function from scran.

Usage

```
differential_analysis_scExp(
  scExp,
  de_type = c("one_vs_rest_fast", "one_vs_rest", "pairwise", "custom")[1],
  by = "cell_cluster",
  method = "wilcox",
  block = NULL,
  group = NULL,
  ref = NULL,
  prioritize_genes = nrow(scExp) > 20000,
  max_distanceToTSS = 1000,
  progress = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

scExp	A SingleCellExperiment object containing consclust with selected number of cluster.
de_type	Type of comparisons. Either 'one_vs_rest', to compare each cluster against all others, or 'pairwise' to make 1 to 1 comparisons. ('one_vs_rest')
by	= A character specifying the column of the object containing the groups of cells to compare. Exclusive with de_type == custom
method	Differential testing method, either 'wilcox' for Wilcoxon non- parametric testing or 'neg.binomial' for edgerGLM based testing. ('wilcox')
block	Use batches as blocking factors ? If TRUE, block will be taken as the column "batch_id" from the SCE. Cells will be compared only within samples belonging to the same batch.
group	If de_type = "custom", the sample / cluster of interest as a one- column data.frame. The name of the column is the group name and the values are character either cluster ("C1", "C2", ...) or sample_id.
ref	If de_type = "custom", the sample / cluster of reference as a one- column data.frame. The name of the column is the group name and the values are character either cluster ("C1", "C2", ...) or sample_id.
prioritize_genes	First filter by loci being close to genes ? E.g. for differential analysis, it is more relevant to keep features close to genes
max_distanceToTSS	If prioritize_genes is TRUE, the maximum distance to consider a feature close to a gene.
progress	A shiny Progress instance to display progress bar.
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Details

This functions takes as input a SingleCellExperiment object with consclust, the type of comparison, either 'one_vs_rest' or 'pairwise', the adjusted p-value threshold (qval.th) and the fold-change threshold (logFC.th). It outputs a SingleCellExperiment object containing a differential list.

Value

Returns a SingleCellExperiment object containing a differential list.

Examples

```
data("scExp")
scExp_cf = differential_analysis_scExp(scExp)
```

distPearson *distPearson*

Description

distPearson

Usage

```
distPearson(m)
```

Arguments

m	A matrix

Value

A dist object

enrichmentTest *enrichmentTest*

Description

enrichmentTest

Usage

```
enrichmentTest(gene.sets, mylist, possibleIds, sep = ";", silent = FALSE)
```

Arguments

gene.sets	A list of reference gene sets
mylist	A list of genes to test
possibleIds	All existing genes
sep	Separator used to collapse genes
silent	Silent mode ?

Value

A dataframe with the gene sets and their enrichment p.value

enrich_TF_ChEA3_genes *Find the TF that are enriched in the differential genes using ChEA3 API*

Description

Find the TF that are enriched in the differential genes using ChEA3 API

Usage

```
enrich_TF_ChEA3_genes(genes)
```

Arguments

genes	A character vector with the name of genes to enrich for TF.

Value

Returns a SingleCellExperiment object containing list of enriched Gene Sets for each cluster, either in depleted features, enriched features or simply differential features (both).

References

Keenan AB, Torre D, Lachmann A, Leong AK, Wojciechowicz M, Utti V, Jagodnik K, Kropiwnicki E, Wang Z, Ma'ayan A (2019) ChEA3: transcription factor enrichment analysis by orthogonal omics integration. Nucleic Acids Research. doi: 10.1093/nar/gkz446 +

Examples

```
data(scExp)
enrich_TF_ChEA3_genes(head(unlist(strsplit(SummarizedExperiment::rowData(scExp)$Gene, split = ",", fixed = TRUE)
```

enrich_TF_ChEA3_scExp *Find the TF that are enriched in the differential genes using ChEA3 database*

Description

Find the TF that are enriched in the differential genes using ChEA3 database

Usage

```
enrich_TF_ChEA3_scExp(
  scExp,
  qval.th = 0.01,
  logFC.th = 1,
  min.percent = 0.01,
  peak_distance = 1000,
  use_peaks = FALSE,
  progress = NULL,
  verbose = TRUE
)
```

Arguments

scExp	A SingleCellExperiment object containing list of differential features.
qval.th	Adjusted p-value threshold to define differential features. (0.01)
logFC.th	Fold change threshold to define differential features. (1)
min.percent	Minimum fraction of cells having the feature active to consider it as significantly differential. (0.01)
peak_distance	Maximum distanceToTSS of feature to gene TSS to consider associated, in bp. (1000)
use_peaks	Use peak calling method (must be calculated beforehand). (FALSE)
progress	A shiny Progress instance to display progress bar.
verbose	A logical to print message or not. (TRUE)

Value

Returns a SingleCellExperiment object containing list of enriched Gene Sets for each cluster, either in depleted features, enriched features or simply differential features (both).

Examples

```
data("scExp")

scExp = enrich_TF_ChEA3_scExp(
 scExp,
 qval.th = 0.01,
 logFC.th = 1,
 min.percent = 0.01)
```

exclude_features_scExp *Remove specific features (CNA, repeats)*

Description

Remove specific features (CNA, repeats)

Usage

```
exclude_features_scExp(
  scExp,
  features_to_exclude,
  by = "region",
  verbose = TRUE
)
```

Arguments

scExp	A SingleCellExperiment object.
features_to_exclude	A GenomicRanges object or data.frame containing genomic regions or features to exclude or path towards a BED file containing the features to exclude.
by	Type of features. Either 'region' or 'feature_name'. If 'region', will look for genomic coordinates in columns 1-3 (chr,start,stop). If 'feature_name', will look for a genes in first column. ('region')
verbose	(TRUE)

Value

A SingleCellExperiment object without features to exclude.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
features_to_exclude = data.frame(chr=c("chr4","chr7","chr17"),
start=c(50000,8000000,2000000),
end=c(100000,16000000,2500000))
features_to_exclude = as(features_to_exclude,"GRanges")
scExp = exclude_features_scExp(scExp,features_to_exclude)
scExp
```

feature_annotation_scExp *Add gene annotations to features*

Description

Add gene annotations to features

Usage

```
feature_annotation_scExp(scExp, ref = "hg38", reference_annotation = NULL)
```

Arguments

scExp	A SingleCellExperiment object.
ref	Reference genome. Either 'hg38', 'mm10' or 'ce11'. ('hg38')
reference_annotation	A data.frame containing gene (or else) annotation with genomic coordinates.

Value

A SingleCellExperiment object with annotated rowData.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp = feature_annotation_scExp(scExp)
head(SummarizedExperiment::rowRanges(scExp))

# Mouse
raw = create_scDataset_raw(ref = "mm10")
scExp = create_scExp(raw$mat, raw$annot)
scExp = feature_annotation_scExp(scExp,ref="mm10")
head(SummarizedExperiment::rowRanges(scExp))
```

filter_correlated_cell_scExp *Filter lowly correlated cells*

Description

Remove cells that have a correlation score lower than what would be expected by chance with other cells.

Usage

```
filter_correlated_cell_scExp(scExp, random_iter = 5,
corr_threshold = 99, percent_correlation = 1,
downsample = 2500, verbose = TRUE, n_process = 250,
BPPARAM = BiocParallel::bpparam())
```

Arguments

scExp	A SingleCellExperiment object containing 'Cor', a correlation matrix, in reducedDims.
random_iter	Number of random matrices to create to calculate random correlation scores. (50)
corr_threshold	Quantile of random correlation score above which a cell is considered to be 'correlated' with another cell. (99)
percent_correlation	Percentage of the cells that any cell must be 'correlated' to in order to not be filtered. (1)
downsample	Number of cells to calculate correlation filtering threshold ? (2500)
verbose	Print messages ? (TRUE)
n_process	Number of cell to proceed at a time. Increase this number to increase speed at memory cost
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Details

This functions takes as input a SingleCellExperiment object that must have correlation matrix calculated and outputs a SingleCellExperiment object without lowly correlated cells. TSNE is recalculated.

Value

Returns a SingleCellExperiment object without lowly correlated cells. The calculated correlation score limit threshold is saved in metadata.

Examples

```
data("scExp")
dim(scExp)
scExp_cf = filter_correlated_cell_scExp(scExp,
corr_threshold = 99, percent_correlation = 1)
dim(scExp_cf)
```

filter_genes_with_refined_peak_annotation *Filter genes based on peak calling refined annotation*

Description

Filter genes based on peak calling refined annotation

Usage

```
filter_genes_with_refined_peak_annotation(
  refined_annotation,
  peak_distance,
  signific,
  over,
  under
)
```

Arguments

refined_annotation	A data.frame containing each gene distance to real peak
peak_distance	Minimum distance to an existing peak to accept a given gene
signific	Indexes of all significantly differential genes
over	Indexes of all significantly overexpressed genes
under	Indexes of all significantly underexpressed genes

Value

List of significantly differential, overexpressed and underexpressed genes close enough to existing peaks

filter_scExp *Filter cells and features*

Description

Function to filter out cells & features from SingleCellExperiment based on total count per cell, number of cells 'ON' in features and top covered cells that might be doublets.

Usage

```
filter_scExp(
  scExp,
  min_cov_cell = 1600,
  quant_removal = 95,
  min_count_per_feature = 10,
  verbose = TRUE
)
```

Arguments

scExp	A SingleCellExperiment object.
min_cov_cell	Minimum counts for each cell. (1600)
quant_removal	Centile of cell counts above which cells are removed. (95)
min_count_per_feature	Minimum number of reads per feature (10).
verbose	(TRUE)

Value

Returns a filtered SingleCellExperiment object.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp. = filter_scExp(scExp)

# No feature filtering (all features are valuable)
scExp. = filter_scExp(scExp,min_count_per_feature=30)

# No cell filtering (all features are valuable)
scExp. = filter_scExp(scExp,min_cov_cell=0,quant_removal=100)
```

---

find_clusters_louvain_scExp

Build SNN graph and find cluster using Louvain Algorithm

Description

Build SNN graph and find cluster using Louvain Algorithm

Usage

```
find_clusters_louvain_scExp(
  scExp,
  k = 10,
  resolution = 1,
  use.dimred = "PCA",
  type = c("rank", "number", "jaccard")[3],
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

scExp	A SingleCellExperiment with PCA calculated
k	An integer scalar specifying the number of nearest neighbors to consider during graph construction.
resolution	A numeric specifying the resolution of clustering to pass to igraph::cluster_louvain function.
use.dimred	A string specifying the dimensionality reduction to use.
type	A string specifying the type of weighting scheme to use for shared neighbors.
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A SingleCellExperiment containing the vector of clusters (named C1, C2 ....)

Examples

```
data('scExp')

scExp = find_clusters_louvain_scExp(scExp, k = 10)
```

---

find_top_features *Find most covered features*

Description

Find the top most covered features that will be used for dimensionality reduction. Optionally remove non-top features.

Usage

```
find_top_features(
  scExp,
  n = 20000,
  keep_others = FALSE,
  prioritize_genes = FALSE,
  max_distanceToTSS = 10000,
  verbose = TRUE
)
```

Arguments

scExp	A SingleCellExperiment.
n	Either an integer indicating the number of top covered regions to find or a character vector of the top percentile of features to keep (e.g. 'q20' to keep top 20% features).
keep_others	Logical indicating if non-top regions are to be removed from the SCE or not (FALSE).
prioritize_genes	First filter by loci being close to genes ? E.g. for differential analysis, it is more relevant to keep features close to genes
max_distanceToTSS	If prioritize_genes is TRUE, the maximum distance to consider a feature close to a gene.
verbose	Print ?

Value

A SCE with top features

Examples

```
data(scExp)
scExp_top = find_top_features(scExp, n = 4000, keep_others = FALSE)
```

---

generate_analysis *Generate a complete ChromSCape analysis*

Description

Generate a complete ChromSCape analysis

Usage

```
generate_analysis(input_data_folder,
analysis_name = "Analysis_1",
output_directory = "./",
input_data_type = c("scBED", "DenseMatrix", "SparseMatrix", "scBAM")[1],
feature_count_on = c("bins","genebody","peaks")[1],
feature_count_parameter = 50000,
rebin_sparse_matrix = FALSE,
ref_genome = c("hg38","mm10", "ce11")[1],
run = c("filter", "CNA","cluster", "consensus", "coverage",
"DA", "GSA", "report")[c(1,3,5,6,7,8)],
min_reads_per_cell = 1000,
max_quantile_read_per_cell = 99,
n_top_features = 40000,
norm_type = "CPM",
subsample_n = NULL,
exclude_regions = NULL,
n_clust = NULL,
corr_threshold = 99,
percent_correlation = 1,
maxK = 10,
qval.th = 0.1,
logFC.th = 1,
enrichment_qval = 0.1,
doBatchCorr  = FALSE,
batch_sels  = NULL,
control_samples_CNA = NULL,
genes_to_plot = c("Krt8","Krt5","Tgfb1", "Foxq1", "Cdkn2b",
                  "Cdkn2a", "chr7:15000000-20000000")
)
```

Arguments

`input_data_folder`
: Directory containing the input data.

`analysis_name`
: Name given to the analysis.

`output_directory`
: Directory where to create the analysis and the HTML report.

`input_data_type`
: The type of input data.

`feature_count_on`
: For raw data type, on which features to count the cells.

`feature_count_parameter`
: Additional parameter corresponding to the 'feature_count_on' parameter. E.g. for 'bins' must be a numeric, e.g. 50000, for 'peaks' must be a character containing path towards a BED peak file.

rebin_sparse_matrix
: A boolean specifying if the SparseMatrix should be rebinned on features (see feature_count_on and feature_count_parameter).

ref_genome
: The genome of reference.

run
: What steps to run. By default runs everything. Some steps are required in order to run downstream steps.

min_reads_per_cell
: Minimum number of reads per cell.

max_quantile_read_per_cell
: Upper quantile above which to consider cells doublets.

n_top_features
: Number of features to keep in the analysis.

norm_type
: Normalization type.

subsample_n
: Number of cells per condition to downsample to, for performance principally.

exclude_regions
: Path towards a BED file containing CNA to exclude from the analysis (optional).

n_clust
: Number of clusters to force choice of clusters.

corr_threshold
: Quantile of correlation above which two cells are considered as correlated.

percent_correlation
: Percentage of the total cells that a cell must be correlated with in order to be kept in the analysis.

maxK
: Upper cluster number to rest for ConsensusClusterPlus.

qval.th
: Adjusted p-value below which to consider features differential.

logFC.th
: Log2-fold-change above/below which to consider a feature depleted/enriched.

enrichment_qval
: Adjusted p-value below which to consider a gene set as significantly enriched in differential features.

doBatchCorr
: Logical indicating if batch correction using fastMNN should be run.

batch_sels
: If doBatchCorr is TRUE, a named list containing the samples in each batch.

control_samples_CNA
: If running CopyNumber Analysis, a character vector of the sample names that are 'normal'.

genes_to_plot
: A character vector containing genes of interest of which to plot the coverage.

Value

Creates a ChromSCape-readable directory and saved objects, as well as a multi-tabbed HTML report resuming the analysis.

Examples

```
## Not run:
generate_analysis("/path/to/data/", "Analysis_1")

## End(Not run)
```

generate_count_matrix *Generate count matrix*

Description

Generate count matrix

Usage

```
generate_count_matrix(cells, features, sparse, cell_names, feature_names)
```

Arguments

cells	Number of cells
features	Number of features
sparse	Is matrix sparse ?
cell_names	Cell names
feature_names	Feature names

Value

A matrix or a sparse matrix

generate_coverage_tracks *Generate cell cluster pseudo-bulk coverage tracks*

Description

Generate cell cluster pseudo-bulk coverage tracks. First, scBED files are concatenated into cell clusters contained in the 'by' column of your SingleCellExperiment object. To do so, for each sample in the given list, the barcodes of each cluster are grepped and BED files are merged into pseudo-bulk of clusters (C1,C2...). Two cells from different can have the same barcode ID as cell affectation is done sample by sample. Then coverage of pseudo-bulk BED files is calculated by averaging & smoothing reads on small genomic window (150bp per default). The pseudo bulk BED and BigWigs coverage tracks are writtend to the output directory. This functionality is not available on Windows as it uses the 'cat' and 'gzip' utilities from Unix OS.

Usage

```
generate_coverage_tracks(
  scExp_cf,
  input,
  odir,
  format = "scBED",
  ref_genome = c("hg38", "mm10", "ce11")[1],
  bin_width = 150,
  n_smoothBin = 5,
  read_size = 101,
  quantile_for_peak_calling = 0.85,
  by = "cell_cluster",
  progress = NULL
)
```

Arguments

scExp_cf	A SingleCellExperiment with cluster selected. (see choose_cluster_scExp). It is recommended having a minimum of ~100 cells per cluster in order to obtain smooth tracks.
input	Either a named list of character vector of path towards single-cell BED files or a sparse raw matrix of small bins («500bp). If a named list specifying scBED the names MUST correspond to the 'sample_id' column in your SingleCellExperiment object. The single-cell BED files names MUST match the barcode names in your SingleCellExperiment (column 'barcode'). The scBED files can be gzipped or not.
odir	The output directory to write the cumulative BED and BigWig files.
format	File format, either "raw_mat", "BED" or "BAM"
ref_genome	The genome of reference, used to constrain to canonical chromosomes. Either 'hg38' or 'mm10'. 'hg38' per default.
bin_width	The width of the bin to create the coverage track. The smaller the greater the resolution & runtime. Default to 150.
n_smoothBin	Number of bins left & right to average ('smooth') the signal on. Default to 5.
read_size	The estimated size of reads. Default to 101.
quantile_for_peak_calling	The quantile to define the threshold above which signal is considered as a peak.
by	A character specifying a categoricla column of scExp_cf metadata by which to group cells and generate coverage tracks and peaks.
progress	A Progress object for Shiny. Default to NULL.

Value

Generate coverage tracks (.bigwig) for each group in the SingleCellExperiment "by" column.

Examples

```
## Not run:
data(scExp)
input_files_coverage = list(
  "scChIP_Jurkat_K4me3" = paste0("/path/to/",scExp$barcode[1:51],".bed"),
  "scChIP_Ramos_K4me3" = paste0("/path/to/",scExp$barcode[52:106],".bed")
)
generate_coverage_tracks(scExp, input_files_coverage, "/path/to/output",
ref_genome = "hg38")

## End(Not run)
```

---

generate_feature_names

Generate feature names

Description

Generate feature names

Usage

```
generate_feature_names(featureType, ref, features)
```

Arguments

featureType	Type of feature
ref	Reference genome
features	Number of features to generate

Value

A character vector of feature names

---

generate_report

From a ChromSCape analysis directory, generate an HTML report.

Description

From a ChromSCape analysis directory, generate an HTML report.

Usage

```
generate_report(
  ChromSCape_directory,
  prefix = NULL,
 run = c("filter", "CNA", "cluster", "consensus", "peak_call", "coverage", "DA", "GSA",
    "report")[c(1, 3, 6, 7, 8, 9)],
  genes_to_plot = c("Krt8", "Krt5", "Tgfb1", "Foxq1", "Cdkn2b", "Cdkn2a",
    "chr7:15000000-20000000"),
  control_samples_CNA = NULL
)
```

Arguments

ChromSCape_directory	Path towards the ChromSCape directory of which you want to create the report. The report will be created at the root of this directory.
prefix	Name of the analysis with the filtering parameters (e.g. Analysis_3000_100000_99_uncorrected). You will find the prefix in the Filtering_Normalize_Reduce subfolder.
run	Which steps to report ("filter", "CNA","cluster", "consensus", "peak_call", "coverage", "DA", "GSA", "report"). Only indicate steps that were done in the analysis. By default do not report CNA, consensus and peak calling.
genes_to_plot	For the UMAP, which genes do you want to see in the report.
control_samples_CNA	If running the Copy Number Alteration (CNA) part, which samples are the controls

Value

Generate an HTML report at the root of the analysis directory.

Examples

```
## Not run:
generate_analysis("/path/to/data/", "Analysis_1")

## End(Not run)
```

---

gene_set_enrichment_analysis_scExp

Runs Gene Set Enrichment Analysis on genes associated with differential features

---

Description

This function takes previously calculated differential features and runs hypergeometric test to look for enriched gene sets in the genes associated with differential features, for each cell cluster. This functions takes as input a SingleCellExperiment object with consclust, the type of comparison, either 'one_vs_rest' or 'pairwise', the adjusted p-value threshold (qval.th) and the fold-change threshold (logFC.th). It outputs a SingleCellExperiment object containing a differential list.

Usage

```
gene_set_enrichment_analysis_scExp(
  scExp,
  enrichment_qval = 0.1,
  ref = "hg38",
  GeneSets = NULL,
  GeneSetsDf = NULL,
  GenePool = NULL,
  qval.th = 0.01,
  logFC.th = 1,
  min.percent = 0.01,
  peak_distance = 1000,
  use_peaks = FALSE,
 GeneSetClasses = c("c1_positional", "c2_curated", "c3_motif", "c4_computational",
    "c5_GO", "c6_oncogenic", "c7_immunologic", "hallmark"),
  progress = NULL
)
```

Arguments

scExp	A SingleCellExperiment object containing list of differential features.
enrichment_qval	Adjusted p-value threshold for gene set enrichment. (0.1)
ref	A reference annotation, either 'hg38', 'mm10', 'ce11'. ('hg38')
GeneSets	A named list of gene sets. If NULL will automatically load MSigDB list of gene sets for specified reference genome. (NULL)
GeneSetsDf	A dataframe containing gene sets & class of gene sets. If NULL will automatically load MSigDB dataframe of gene sets for specified reference genome. (NULL)
GenePool	The pool of genes to run enrichment in. If NULL will automatically load Gencode list of genes fro specified reference genome. (NULL)
qval.th	Adjusted p-value threshold to define differential features. (0.01)
logFC.th	Fold change threshold to define differential features. (1)
min.percent	Minimum fraction of cells having the feature active to consider it as significantly differential. (0.01)
peak_distance	Maximum distanceToTSS of feature to gene TSS to consider associated, in bp. (1000)
use_peaks	Use peak calling method (must be calculated beforehand). (FALSE)

GeneSetClasses	Which classes of MSIGdb to look for.
progress	A shiny Progress instance to display progress bar.

Value

Returns a SingleCellExperiment object containing list of enriched Gene Sets for each cluster, either in depleted features, enriched features or simply differential features (both).

Examples

```
data("scExp")

#Usually recommanding qval.th = 0.01 & logFC.th = 1 or 2
## Not run: scExp_cf = gene_set_enrichment_analysis_scExp(scExp,
 qval.th = 0.4, logFC.th = 0.3)
## End(Not run)
```

---

getExperimentNames *Get experiment names from a SingleCellExperiment*

Description

Get experiment names from a SingleCellExperiment

Usage

```
getExperimentNames(scExp)
```

Arguments

scExp	A SingleCellExperiment with named mainExp and altExps.

Value

Character vector of unique experiment names

Examples

```
data(scExp)
getExperimentNames(scExp)
```

`getMainExperiment` *Get Main experiment of a SingleCellExperiment*

Description

Get Main experiment of a SingleCellExperiment

Usage

```
getMainExperiment(scExp)
```

Arguments

`scExp`	A SingleCellExperiment with named mainExp and altExps.

Value

The swapped SingleCellExperiment towards "main" experiment

Examples

```
data(scExp)
getMainExperiment(scExp)
```

`get_color_dataframe_from_input` *Get color dataframe from shiny::colorInput*

Description

Get color dataframe from shiny::colorInput

Usage

```
get_color_dataframe_from_input(
  input,
  levels_selected,
  color_by = c("sample_id", "total_counts"),
  input_id_prefix = "color_"
)
```

Arguments

input	Shiny input object
levels_selected	Names of the features
color_by	Which feature color to retrieve
input_id_prefix	Prefix in front of the feature names

Value

A data.frame with the feature levels and the colors of each level of this feature.

---

get_cyto_features *Map features onto cytobands*

Description

Map the features of a SingleCellExperiment onto the cytobands of a given genome. Some features might not be mapped to any cytobands (e.g. if they are not in the canconical chromosomes), and are removed from the returned object.

Usage

```
get_cyto_features(scExp, ref_genome = c("hg38", "mm10", "ce11")[1])
```

Arguments

scExp	A SingleCellExperiment with genomic coordinate as features (peaks or bins)
ref_genome	Reference genome ('hg38' or 'mm10')

Details

The cytobands are an arbitrary cutting of the genome that dates back to staining metaphase chromosomes with Giemsa.

Value

A data.frame of the SCE features with their corresponding cytoband name

Examples

```
data("scExp")
matching_cyto = get_cyto_features(scExp, ref_genome="hg38")
```

`get_genomic_coordinates` *Get SingleCellExperiment's genomic coordinates*

Description

Get SingleCellExperiment's genomic coordinates

Usage

```
get_genomic_coordinates(scExp)
```

Arguments

`scExp`	A SingleCellExperiment object.

Value

A GRanges object of genomic coordinates.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
feature_GRanges = get_genomic_coordinates(scExp)
```

`get_most_variable_cyto` *Retrieve the cytobands with the most variable fraction of reads*

Description

Given a SingleCellExperiment object with the slot "cytoBand" containing the fraction of reads in each cytoband, calculates the variance of each cytoband and returns a data.frame with the top variables cytobands. Most cytobands are expected to be unchanged between normal and tumor samples, therefore focusing on the top variable cytobands enable to focus on the most interseting regions.

Usage

```
get_most_variable_cyto(scExp, top = 50)
```

Arguments

`scExp`	A SingleCellExperiment with "cytoBand" reducedDim slot filled.
`top`	Number of cytobands to return (50).

Value

A data.frame of the top variable cytoBands and their variance

Examples

```
data("scExp")
scExp = calculate_cyto_mat(scExp, ref_genome="hg38")
get_most_variable_cyto(scExp, top=50)
```

`get_pathway_mat_scExp` *Get pathway matrix*

Description

Get pathway matrix

Usage

```
get_pathway_mat_scExp(
  scExp,
  pathways,
  max_distanceToTSS = 1000,
  ref = "hg38",
 GeneSetClasses = c("c1_positional", "c2_curated", "c3_motif", "c4_computational",
    "c5_GO", "c6_oncogenic", "c7_immunologic", "hallmark"),
  progress = NULL
)
```

Arguments

`scExp`	A SingleCellExperiment
`pathways`	A character vector specifying the pathways to retrieve the cell count for.
`max_distanceToTSS`	Numeric. Maximum distance to a gene's TSS to consider a region linked to a gene. (1000)#' @param ref
`ref`	Reference genome, either mm10 or hg38
`GeneSetClasses`	Which classes of MSIGdb to load
`progress`	A shiny Progress instance to display progress bar.

Value

A matrix of cell to pathway

Examples

```
data(scExp)
mat = get_pathway_mat_scExp(scExp, pathways = "KEGG_T_CELL_RECEPTOR_SIGNALING_PATHWAY")
```

`gg_fill_hue` *gg_fill_hue*

Description

gg_fill_hue

Usage

```
gg_fill_hue(n)
```

Arguments

`n`	num hues

Value

A color in HEX format

`groupMat` *groupMat*

Description

groupMat

Usage

```
groupMat(mat = NA, margin = 1, groups = NA, method = "mean")
```

Arguments

`mat`	A matrix
`margin`	By row or columns ?
`groups`	Groups
`method`	Method to group

Value

A grouped matrix

`H1proportion` *H1proportion*

Description

H1proportion

Usage

```
H1proportion(pv = NA, lambda = 0.5)
```

Arguments

`pv`	P.value vector
`lambda`	Lambda value

Value

H1 proportion value

`has_genomic_coordinates` *Does SingleCellExperiment has genomic coordinates in features ?*

Description

Does SingleCellExperiment has genomic coordinates in features ?

Usage

```
has_genomic_coordinates(scExp)
```

Arguments

`scExp`	A SingleCellExperiment object

Value

TRUE or FALSE

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
has_genomic_coordinates(scExp)
raw_genes = create_scDataset_raw(featureType="gene")
scExp_gene = create_scExp(raw_genes$mat, raw_genes$annot)
has_genomic_coordinates(scExp_gene)
```

`hclustAnnotHeatmapPlot`

hclustAnnotHeatmapPlot

Description

hclustAnnotHeatmapPlot

Usage

```
hclustAnnotHeatmapPlot(
  x = NULL,
  hc = NULL,
  hmColors = NULL,
  anocol = NULL,
  xpos = c(0.1, 0.9, 0.114, 0.885),
  ypos = c(0.1, 0.5, 0.5, 0.6, 0.62, 0.95),
  dendro.cex = 1,
  xlab.cex = 0.8,
  hmRowNames = FALSE,
  hmRowNames.cex = 0.5
)
```

Arguments

`x`	A correlation matrix
`hc`	An hclust object
`hmColors`	A color palette
`anocol`	A matrix of colors
`xpos`	Xpos
`ypos`	Ypos
`dendro.cex`	Size of denro names
`xlab.cex`	Size of x label
`hmRowNames`	Write rownames ?
`hmRowNames.cex`	Size of rownames ?

Value

A heatmap

---

hg38.chromosomes *Data.frame of chromosome length - hg38*

Description

This data frame provides the length of each "canonical" chromosomes of Homo Sapiens genome build hg38.

Usage

```
data("hg38.chromosomes")
```

Format

hg38.chromosomes - a data frame with 24 rows and 3 variables:

chr Chromosome - character

start Start of the chromosome (bp) - integer

end End of the chromosome (bp) - integer

Value

hg38.chromosomes - a data frame with 24 rows and 3 variables.

---

hg38.cytoBand *Data.frame of cytoBandlocation - hg38*

Description

This data frame provides the location of each cytoBands of Homo Sapiens genome build hg38.

Usage

```
data("hg38.cytoBand")
```

Format

hg38.cytoBand - a data frame with 862 cytobands and 4 variables:

chr Chromosome - character

start Start of the chromosome (bp) - integer

end End of the chromosome (bp) - integer

cytoBand Name of the cytoBand - character

Value

hg38.cytoBand - a data frame with 862 cytobands and 4 variables.

---

hg38.GeneTSS *Data.frame of gene TSS - hg38*

Description

This dataframe was extracted from Gencode v25 and report the Transcription Start Site of each gene in the Homo Sapiens genome build hg38.

Usage

```
data("hg38.GeneTSS")
```

Format

hg38.GeneTSS - a data frame with 32,937 genes and 5 variables:

chr Chromosome - character

start Start of the gene (TSS) - integer

end End of the gene - integer

Gene Gene symbol - character

Strand Strand - character

Value

hg38.GeneTSS - a data frame with 32,937 genes and 5 variables

---

imageCol *imageCol*

Description

imageCol

Usage

```
imageCol(
  matcol = NULL,
  strat = NULL,
  xlab.cex = 0.5,
  ylab.cex = 0.5,
  drawLines = c("none", "h", "v", "b")[1],
  ...
)
```

Arguments

matcol	A matrix of colors
strat	Strat
xlab.cex	X label size
ylab.cex	Y label size
drawLines	Draw lines ?
...	Additional parameters

Value

A rectangular image

---

import_count_input_files

Import and count input files depending on their format

Description

Import and count input files depending on their format

Usage

```
import_count_input_files(
  files_dir_list,
  file_type,
  which,
  ref,
  verbose,
  progress,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

files_dir_list	A named list of directories containing the input files.
file_type	Input file type.
which	A GRanges object of features.
ref	Reference genome.
verbose	Print ?
progress	A progress object for Shiny.
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A list with the feature indexes data.frame containing non-zeroes entries in the count matrix and the cell names

---

`import_scExp` *Read single-cell matrix(ces) into scExp*

Description

Combine one or multiple matrices together to create a sparse matrix and cell annotation data.frame.

Usage

```
import_scExp(file_paths, remove_pattern = "", temp_path = NULL)
```

Arguments

`file_paths`	A character vector of file names towards single cell epigenomic matrices (features x cells) (must be .txt / .tsv)
`remove_pattern`	A string pattern to remove from the sample names. Can be a regexp.
`temp_path`	In case matrices are stored in temporary folder, a character vector of path towards temporary files. (NULL)

Value

A list containing:

- datamatrix: a sparseMatrix of features x cells
- annot_raw: an annotation of cells as data.frame

Examples

```
mat1 = mat2 = create_scDataset_raw()$mat
tmp1 = tempfile(fileext = ".tsv")
tmp2 = tempfile(fileext = ".tsv")
write.table(as.matrix(mat1),file=tmp1,sep = "\t",
row.names = TRUE,col.names = TRUE,quote = FALSE)
write.table(as.matrix(mat2),file=tmp2, sep = "\t",
row.names = TRUE,col.names = TRUE,quote = FALSE)
file_paths = c(tmp1,tmp2)
out = import_scExp(file_paths)
```

`index_peaks_barcodes_to_matrix_indexes` *Read index-peaks-barcodes trio files on interval to create count indexes*

Description

Read index-peaks-barcodes trio files on interval to create count indexes

Usage

```
index_peaks_barcodes_to_matrix_indexes(
  feature_file,
  matrix_file,
  barcode_file,
  binarize = FALSE
)
```

Arguments

`feature_file`	A file containing the features genomic locations
`matrix_file`	A file containing the indexes of non-zeroes values and their value (respectively i,j,x,see sparseMatrix)
`barcode_file`	A file containing the barcode ids
`binarize`	Binarize matrix ?

Value

A list containing a "feature index" data.frame, name_cells, and a region GenomicRange object used to form the sparse matrix

`inter_correlation_scExp` *Calculate inter correlation between cluster or samples*

Description

Calculate inter correlation between cluster or samples

Usage

```
inter_correlation_scExp(
  scExp_cf,
  by = c("sample_id", "cell_cluster")[1],
  reference_group = unique(scExp_cf[[by]])[1],
  other_groups = unique(scExp_cf[[by]]),
  fullCor = TRUE
)
```

Arguments

scExp_cf	A SingleCellExperiment
by	On which feature to calculate correlation ("sample_id" or "cell_cluster")
reference_group	Reference group to calculate correlation with. Must be in accordance with "by".
other_groups	Groups on which to calculate correlation (can contain multiple groups, and also reference_group). Must be in accordance with "by".
fullCor	A logical specifying if the correlation matrix was calculated on the entire set of cells (TRUE).

Value

A data.frame of average inter-correlation of cells in other_groups with cells in reference_group

Examples

```
data(scExp)
inter_correlation_scExp(scExp)
```

---

intra_correlation_scExp

Calculate intra correlation between cluster or samples

Description

Calculate intra correlation between cluster or samples

Usage

```
intra_correlation_scExp(
  scExp_cf,
  by = c("sample_id", "cell_cluster")[1],
  fullCor = TRUE
)
```

Arguments

scExp_cf	A SingleCellExperiment
by	On which feature to calculate correlation ("sample_id" or "cell_cluster")
fullCor	Logical specifying if the correlation matrix was run on the entire number of cells or on a subset.

Value

A data.frame of cell average intra-correlation

Examples

```
data(scExp)
intra_correlation_scExp(scExp, by = "sample_id")
intra_correlation_scExp(scExp, by = "cell_cluster")
```

---

launchApp *Launch ChromSCape*

Description

Main function to launch ChromSCape in your favorite browser. You can pass additional parameters that you would pass to shiny::runApp (runApp)

Usage

```
launchApp(launch.browser = TRUE, ...)
```

Arguments

launch.browser	Wether to launch browser or not
...	Additional parameters passed to runApp

Value

Launches the shiny application

Examples

```
## Not run:
launchApp()

## End(Not run)
```

load_MSIGdb *Load and format MSIGdb pathways using msigdbr package*

Description

Load and format MSIGdb pathways using msigdbr package

Usage

```
load_MSIGdb(
  ref,
 GeneSetClasses = c("c1_positional", "c2_curated", "c3_motif", "c4_computational",
    "c5_GO", "c6_oncogenic", "c7_immunologic", "hallmark")
)
```

Arguments

ref	Reference genome, either mm10 or hg38
GeneSetClasses	Which classes of MSIGdb to load

Value

A list containing the GeneSet (list), GeneSetDf (data.frame) and GenePool character vector of all possible genes

merge_MACS2_peaks *Merge peak files from MACS2 peak caller*

Description

Merge peak files from MACS2 peak caller

Usage

```
merge_MACS2_peaks(peak_file, peak_distance_to_merge, min_peak_size = 200, ref)
```

Arguments

peak_file	A character specifying the path towards the peak file (BED or bedGraph format)
peak_distance_to_merge	Maximum distance to merge two peaks
min_peak_size	An integer specifying the minimum size of peaks
ref	Reference genome

Value

Peaks as GRanges

mm10.chromosomes *Data.frame of chromosome length - mm10*

Description

This data frame provides the length of each "canonical" chromosomes of Mus Musculus (Mouse) genome build mm10.

Usage

```
data("mm10.chromosomes")
```

Format

mm10.chromosomes - a data frame with 24 chromosomes and 3 variables:

chr Chromosome - character

start Start of the chromosome (bp) - integer

end End of the chromosome (bp) - integer

Value

mm10.chromosomes - a data frame with 24 chromosomes and 3 variables

mm10.cytoBand *Data.frame of cytoBandlocation - mm10*

Description

This data frame provides the location of each cytoBands of Homo Sapiens genome build mm10.

Usage

```
data("mm10.cytoBand")
```

Format

mm10.cytoBand - a data frame with 403 cytobands and 4 variables:

chr Chromosome - character

start Start of the chromosome (bp) - integer

end End of the chromosome (bp) - integer

cytoBand Name of the cytoBand - character

Value

mm10.cytoBand - a data frame with 403 cytobands and 4 variables.

mm10.GeneTSS *Data.frame of gene TSS - mm10*

Description

This dataframe was extracted from Gencode v25 and report the Transcription Start Site of each gene in the Mus Musculus genome build mm10 (Mouse).

Usage

```
data("mm10.GeneTSS")
```

Format

mm10.GeneTSS - a data frame with 27,916 genes and 5 variables:

chr Chromosome name - character

start Start of the gene (TSS) - integer

end End of the gene - integer

Gene Gene symbol - character

Strand Strand - character

Value

mm10.GeneTSS - a data frame with 27,916 genes and 5 variables

normalize_scExp *Normalize counts*

Description

Normalize counts

Usage

```
normalize_scExp(
  scExp,
  type = c("CPM", "TFIDF", "RPKM", "TPM", "feature_size_only")
)
```

Arguments

scExp	A SingleCellExperiment object.
type	Which normalization to apply. Either 'CPM', 'TFIDF','RPKM', 'TPM' or 'feature_size_only'. Note that for all normalization by size (RPKM, TPM, feature_size_only), the features must have defined genomic coordinates.

Value

A SingleCellExperiment object containing normalized counts. (See ?normcounts())

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp = normalize_scExp(scExp)
head(SingleCellExperiment::normcounts(scExp))
```

---

num_cell_after_cor_filt_scExp

Number of cells before & after correlation filtering

Description

Number of cells before & after correlation filtering

Usage

```
num_cell_after_cor_filt_scExp(scExp, scExp_cf)
```

Arguments

scExp	SingleCellExperiment object before correlation filtering.
scExp_cf	SingleCellExperiment object atfer correlation filtering.

Value

A colored kable with the number of cells per sample before and after filtering for display

Examples

```
data("scExp")
scExp_cf = correlation_and_hierarchical_clust_scExp(scExp)
scExp_cf = filter_correlated_cell_scExp(scExp_cf,
corr_threshold = 99, percent_correlation = 1)
## Not run: num_cell_after_cor_filt_scExp(scExp,scExp_cf)
```

num_cell_after_QC_filt_scExp *Table of cells before / after QC*

Description

Table of cells before / after QC

Usage

```
num_cell_after_QC_filt_scExp(scExp, annot, datamatrix)
```

Arguments

scExp	A SingleCellExperiment object.
annot	A raw annotation data.frame of cells before filtering.
datamatrix	A matrix of cells per regions before filtering.

Value

A formatted kable in HTML.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp_filtered = filter_scExp(scExp)
## Not run:  num_cell_after_QC_filt_scExp(
scExp_filtered,as.data.frame(SingleCellExperiment::colData(scExp)), counts(scExp))
## End(Not run)
```

num_cell_before_cor_filt_scExp *Table of number of cells before correlation filtering*

Description

Table of number of cells before correlation filtering

Usage

```
num_cell_before_cor_filt_scExp(scExp)
```

Arguments

scExp	A SingleCellExperiment Object

Value

A colored kable with the number of cells per sample for display

Examples

```
data("scExp")
## Not run: num_cell_before_cor_filt_scExp(scExp)
```

---

num_cell_in_cluster_scExp

Number of cells in each cluster

Description

Number of cells in each cluster

Usage

```
num_cell_in_cluster_scExp(scExp)
```

Arguments

scExp	A SingleCellExperiment object containing chromatin groups.

Value

A formatted kable of cell assignation to each cluster.

Examples

```
data("scExp")
scExp_cf = correlation_and_hierarchical_clust_scExp(scExp)
scExp_cf = choose_cluster_scExp(scExp_cf,nclust=3,consensus=FALSE)
## Not run: num_cell_in_cluster_scExp(scExp_cf)
```

num_cell_scExp *Table of cells*

Description

Table of cells

Usage

```
num_cell_scExp(annot, datamatrix)
```

Arguments

annot	An annotation of cells. Can be obtain through 'colData(scExp)'.
datamatrix	A matrix of cells per regions before filtering.

Value

A formatted kable in HTML.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
## Not run: num_cell_scExp(SingleCellExperiment::colData(scExp))
```

pca_irlba_for_sparseMatrix
Run sparse PCA using irlba SVD

Description

This function allows to run a PCA using IRLBA Singular Value Decomposition in a fast & memory efficient way. The increamental Lanczos bidiagonalisation algorithm allows to keep the matrix sparse as the "loci" centering is implicit. The function then multiplies by the approximate singular values (svd$d) in order to get more importance to the first PCs proportionnally to their singular values. This step is crucial for downstream approaches, e.g. UMAP or T-SNE.

Usage

```
pca_irlba_for_sparseMatrix(x, n_comp, work = 3 * n_comp)
```

Arguments

x	A sparse normalized matrix (features x cells)
n_comp	The number of principal components to keep
work	Working subspace dimension, larger values can speed convergence at the cost of more memory use.

Value

The rotated data, e.g. the cells x PC column in case of sc data.

---

plot_cluster_consensus_scExp

Plot cluster consensus

Description

Plot cluster consensus score for each k as a bargraph.

Usage

```
plot_cluster_consensus_scExp(scExp)
```

Arguments

scExp	A SingleCellExperiment

Value

The consensus score for each cluster for each k as a barplot

Examples

```
data("scExp")

plot_cluster_consensus_scExp(scExp)
```

plot_correlation_PCA_scExp *Plotting correlation of PCs with a variable of interest*

Description

Plotting correlation of PCs with a variable of interest

Usage

```
plot_correlation_PCA_scExp(
  scExp,
  correlation_var = "total_counts",
  color_by = NULL,
  topPC = 10
)
```

Arguments

scExp	A SingleCellExperiment Object
correlation_var	A string specifying with which numeric variable from colData of scExp to calculate and plot the correlation of each PC with. ('total_counts')
color_by	A string specifying with which categorical variable to color the plot. ('NULL')
topPC	An integer specifying the number of PCs to plot correlation with 10

Value

A ggplot histogram representing the distribution of count per cell

Examples

```
data("scExp")
plot_correlation_PCA_scExp(scExp, topPC = 25)
plot_correlation_PCA_scExp(scExp, color_by = "cell_cluster")
plot_correlation_PCA_scExp(scExp, color_by = "sample_id")
```

`plot_coverage_BigWig` *Coverage plot*

Description

Coverage plot

Usage

```
plot_coverage_BigWig(
  coverages,
  label_color_list,
  peaks = NULL,
  chrom,
  start,
  end,
  ref = "hg38"
)
```

Arguments

`coverages`	A list containing sample coverage as GenomicRanges
`label_color_list`	List of colors, list names are labels
`peaks`	A GRanges object containing peaks location to plot (optional)
`chrom`	Chromosome
`start`	Start
`end`	End
`ref`	Genomic Reference

Value

A coverage plot annotated with genes

Examples

```
data(scExp)
```

plot_differential_summary_scExp
Differential summary barplot

Description

Differential summary barplot

Usage

```
plot_differential_summary_scExp(
  scExp_cf,
  qval.th = 0.01,
  logFC.th = 1,
  min.percent = 0.01
)
```

Arguments

scExp_cf	A SingleCellExperiment object
qval.th	Adjusted p-value threshold. (0.01)
logFC.th	Fold change threshold. (1)
min.percent	Minimum fraction of cells having the feature active to consider it as significantly differential. (0.01)

Value

A barplot summary of differential analysis

Examples

```
data("scExp")
plot_differential_summary_scExp(scExp)
```

plot_differential_volcano_scExp
Volcano plot of differential features

Description

Volcano plot of differential features

Usage

```
plot_differential_volcano_scExp(
  scExp_cf,
  group = "C1",
  logFC.th = 1,
  qval.th = 0.01,
  min.percent = 0.01
)
```

Arguments

scExp_cf	A SingleCellExperiment object
group	A character indicating the group for which to plot the differential volcano plot. ("C1")
logFC.th	Fold change threshold. (1)
qval.th	Adjusted p-value threshold. (0.01)
min.percent	Minimum fraction of cells having the feature active to consider it as significantly differential. (0.01)

Value

A volcano plot of differential analysis of a specific cluster

Examples

```
data("scExp")
plot_differential_volcano_scExp(scExp,"C1")
```

---

plot_distribution_scExp

Plotting distribution of signal

Description

Plotting distribution of signal

Usage

```
plot_distribution_scExp(
  scExp,
  raw = TRUE,
  log10 = FALSE,
  pseudo_counts = 1,
  bins = 150
)
```

Arguments

scExp	A SingleCellExperiment Object
raw	Use raw counts ?
log10	Transform using log10 ?
pseudo_counts	Pseudo-count to add if using log10
bins	Number of bins in the histogram

Value

A ggplot histogram representing the distribution of count per cell

Examples

```
data("scExp")
plot_distribution_scExp(scExp)
```

---

plot_gain_or_loss_barplots

Plot Gain or Loss of cytobands of the most variables cytobands

---

Description

Plot Gain or Loss of cytobands of the most variables cytobands

Plot Gain or Loss of cytobands of the most variables cytobands

Usage

```
plot_gain_or_loss_barplots(scExp, cells = NULL, top = 20)
```


```
plot_gain_or_loss_barplots(scExp, cells = NULL, top = 20)
```


Arguments

scExp	A SingleCellExperiment with "logRatio_cytoBand" reducedDim slot filled. See calculate_logRatio_CNA
cells	Cell IDs of the tumor samples to
top	Number of most variables cytobands to plot

Value

Plot the gains/lost in the selected cells of interest as multiple barplots

Plot the gains/lost in the selected cells of interest as multiple barplots

Examples

```
data("scExp")
scExp = calculate_CNA(scExp,  control_samples = unique(scExp$sample_id)[1],
ref_genome="hg38", quantiles_to_define_gol = c(0.05,0.95))
plot_gain_or_loss_barplots(scExp, cells = scExp$cell_id[which(
scExp$sample_id %in% unique(scExp$sample_id)[2])])


data("scExp")
scExp = calculate_CNA(scExp,  control_samples = unique(scExp$sample_id)[1],
ref_genome="hg38", quantiles_to_define_gol = c(0.05,0.95))
plot_gain_or_loss_barplots(scExp, cells = scExp$cell_id[which(
scExp$sample_id %in% unique(scExp$sample_id)[2])])
```

---

plot_heatmap_scExp *Plot cell correlation heatmap with annotations*

Description

Plot cell correlation heatmap with annotations

Usage

```
plot_heatmap_scExp(
  scExp,
  name_hc = "hc_cor",
 corColors = (grDevices::colorRampPalette(c("royalblue", "white", "indianred1")))(256),
  color_by = NULL,
  downsample = 1000,
  hc_linkage = "ward.D"
)
```

Arguments

scExp	A SingleCellExperiment Object
name_hc	Name of the hclust contained in the SingleCellExperiment object
corColors	A palette of colors for the heatmap
color_by	Which features to add as additional bands on top of plot
downsample	Number of cells to downsample
hc_linkage	A linkage method for hierarchical clustering. See cor. ('ward.D')

Value

A heatmap of cell to cell correlation, grouping cells by hierarchical clustering.

Examples

```
data("scExp")
plot_heatmap_scExp(scExp)
```

`plot_inter_correlation_scExp` *Violin plot of inter-correlation distribution between one or multiple groups and one reference group*

Description

Violin plot of inter-correlation distribution between one or multiple groups and one reference group

Usage

```
plot_inter_correlation_scExp(
  scExp_cf,
  by = c("sample_id", "cell_cluster")[1],
  jitter_by = NULL,
  reference_group = unique(scExp_cf[[by]])[1],
  other_groups = unique(scExp_cf[[by]]),
  downsample = 5000
)
```

Arguments

`scExp_cf`	A SingleCellExperiment
`by`	Color by sample_id or cell_cluster
`jitter_by`	Add jitter points of another layer (cell_cluster or sample_id)
`reference_group`	Character containing the reference group name to calculate correlation from.
`other_groups`	Character vector of the other groups for which to calculate correlation with the reference group.
`downsample`	Downsample for plotting

Value

A violin plot of inter-correlation

Examples

```
data(scExp)
plot_intra_correlation_scExp(scExp)
```

plot_intra_correlation_scExp *Violin plot of intra-correlation distribution*

Description

Violin plot of intra-correlation distribution

Usage

```
plot_intra_correlation_scExp(
  scExp_cf,
  by = c("sample_id", "cell_cluster")[1],
  jitter_by = NULL,
  downsample = 5000
)
```

Arguments

scExp_cf	A SingleCellExperiment
by	Color by sample_id or cell_cluster
jitter_by	Add jitter points of another layer (cell_cluster or sample_id)
downsample	Downsample for plotting

Value

A violin plot of intra-correlation

Examples

```
data(scExp)
plot_intra_correlation_scExp(scExp)
```

plot_most_contributing_features *Plot Top/Bottom most contributing features to PCA*

Description

Plot Top/Bottom most contributing features to PCA

Usage

```
plot_most_contributing_features(
  scExp,
  component = "Component_1",
  n_top_bot = 10
)
```

Arguments

scExp	A SingleCellExperiment containing "PCA" in reducedDims and gene annotation in rowRanges
component	The name of the component of interest
n_top_bot	An integer number of top and bot regions to plot

Details

If a gene TSS is within 10,000bp of the region, the name of the gene(s) will be displayed instead of the region

Value

A barplot of top and bottom features with the largest absolute value in the component of interest

Examples

```
data(scExp)
plot_most_contributing_features(scExp, component = "Component_1")
```

---

plot_percent_active_feature_scExp

Barplot of the % of active cells for a given features

Description

Barplot of the % of active cells for a given features

Usage

```
plot_percent_active_feature_scExp(
  scExp,
  gene,
  by = c("cell_cluster", "sample_id")[1],
  highlight = NULL,
  downsample = 5000,
  max_distanceToTSS = 1000
)
```

Arguments

scExp	A SingleCellExperiment
gene	A character specifying the gene to plot
by	Color violin by cell_cluster or sample_id ("cell_cluster")
highlight	A specific group to highlight in a one vs all fashion
downsample	Downsample for plotting (5000)
max_distanceToTSS	Numeric. Maximum distance to a gene's TSS to consider a region linked to a gene. (1000)

Value

A violin plot of intra-correlation

Examples

```
data(scExp)
plot_percent_active_feature_scExp(scExp, "UBXN10")
```

---

plot_pie_most_contributing_chr

Pie chart of top contribution of chromosomes in the 100 most contributing features to PCA #'

Description

Pie chart of top contribution of chromosomes in the 100 most contributing features to PCA #'

Usage

```
plot_pie_most_contributing_chr(
  scExp,
  component = "Component_1",
  n_top_bot = 100
)
```

Arguments

scExp	A SingleCellExperiment containing "PCA" in reducedDims and gene annotation in rowRanges
component	The name of the component of interest
n_top_bot	An integer number of top and bot regions to plot (100)

Value

A pie chart showing the distribution of chromosomes in the top features with the largest absolute value in the component of interest

Examples

```
data(scExp)
 plot_pie_most_contributing_chr(scExp, component = "Component_1")
```

`plot_reduced_dim_scExp`

Plot reduced dimensions (PCA, TSNE, UMAP)

Description

Plot reduced dimensions (PCA, TSNE, UMAP)

Usage

```
plot_reduced_dim_scExp(
  scExp,
  color_by = "sample_id",
  reduced_dim = c("PCA", "TSNE", "UMAP"),
  select_x = NULL,
  select_y = NULL,
  downsample = 5000,
  transparency = 0.6,
  size = 1,
  max_distanceToTSS = 1000,
  annotate_clusters = "cell_cluster" %in% colnames(colData(scExp)),
  min_quantile = 0.01,
  max_quantile = 0.99
)
```

Arguments

`scExp`	A SingleCellExperiment Object
`color_by`	Character of eature used for coloration. Can be cell metadata ('total_counts', 'sample_id', ...) or a gene name.
`reduced_dim`	Reduced Dimension used for plotting
`select_x`	Which variable to select for x axis
`select_y`	Which variable to select for y axis
`downsample`	Number of cells to downsample
`transparency`	Alpha parameter, between 0 and 1
`size`	Size of the points.

max_distanceToTSS
: The maximum distance to TSS to consider a gene linked to a region. Used only if "color_by" is a gene name.

annotate_clusters
: A logical indicating if clusters should be labelled. The 'cell_cluster' column should be present in metadata.

min_quantile
: The lower threshold to remove outlier cells, as quantile of cell embeddings (between 0 and 0.5).

max_quantile
: The upper threshold to remove outlier cells, as quantile of cell embeddings (between 0.5 and 1).

Value

A ggplot geom_point plot of reduced dimension 2D reprensentation

Examples

```
data("scExp")
plot_reduced_dim_scExp(scExp, color_by = "sample_id")
plot_reduced_dim_scExp(scExp, color_by = "total_counts")
plot_reduced_dim_scExp(scExp, reduced_dim = "UMAP")
plot_reduced_dim_scExp(scExp, color_by = "CD52",  reduced_dim = "UMAP")
```

---

plot_reduced_dim_scExp_CNA

Plot UMAP colored by Gain or Loss of cytobands

Description

Plot UMAP colored by Gain or Loss of cytobands

Usage

```
plot_reduced_dim_scExp_CNA(scExp, cytoBand)
```

Arguments

scExp
: A SingleCellExperiment with "gainOrLoss_cytoBand" reducedDim slot filled. See calculate_gain_or_loss

cytoBand
: Which cytoBand to color cells by

Value

Plot the gains/lost of the cytoband overlayed on the epigenetic UMAP.

Examples

```
data("scExp")
scExp = calculate_CNA(scExp,  control_samples = unique(scExp$sample_id)[1],
ref_genome="hg38", quantiles_to_define_gol = c(0.05,0.95))
plot_reduced_dim_scExp_CNA(scExp, get_most_variable_cyto(scExp)$cytoBand[1])
```

---

plot_top_TF_scExp *Barplot of top TFs from ChEA3 TF enrichment analysis*

Description

Barplot of top TFs from ChEA3 TF enrichment analysis

Usage

```
plot_top_TF_scExp(
  scExp,
  group = unique(scExp$cell_cluster)[1],
  set = c("Differential", "Enriched", "Depleted")[1],
  type = c("Score", "nTargets", "nTargets_over_TF", "nTargets_over_genes")[1],
  n_top = 25
)
```

Arguments

scExp	A SingleCellExperiment
group	A character string specifying the differential group to display the top TFs
set	A character string specifying the set of genes in which the TF were enriched, either 'Differential', 'Enriched' or 'Depleted'.
type	A character string specifying the Y axis of the plot, either the number of differential targets or the ChEA3 integrated mean score. E.g. either "Score", "nTargets", "nTargets_over_TF" for the number of target genes over the total number of genes targeted by the TF or "nTargets_over_genes" for the number of target genes over the number of genes in the gene set.
n_top	An integer specifying the number of top TF to display

Value

A bar plot of top TFs from ChEA3 TF enrichment analysis

Examples

```
data("scExp")

plot_top_TF_scExp(
 scExp,
 group = "C1",
  set = "Differential",
   type = "Score",
    n_top = 10)

plot_top_TF_scExp(
 scExp,
 group = "C1",
  set = "Enriched",
   type = "nTargets_over_genes",
    n_top = 20)
```

`plot_violin_feature_scExp` *Violin plot of features*

Description

Violin plot of features

Usage

```
plot_violin_feature_scExp(
  scExp,
  gene,
  by = c("cell_cluster", "sample_id")[1],
  downsample = 5000,
  max_distanceToTSS = 1000
)
```

Arguments

`scExp`	A SingleCellExperiment
`gene`	A character specifying the gene to plot
`by`	Color violin by cell_cluster or sample_id ("cell_cluster")
`downsample`	Downsample for plotting (5000)
`max_distanceToTSS`	Numeric. Maximum distance to a gene's TSS to consider a region linked to a gene. (1000)

Value

A violin plot of intra-correlation

Examples

```
data(scExp)
plot_violin_feature_scExp(scExp, "UBXN10")
```

`preprocessing_filtering_and_reduction` *Preprocess and filter matrix annotation data project folder to SCE*

Description

Preprocess and filter matrix annotation data project folder to SCE

Usage

```
preprocessing_filtering_and_reduction(
  datamatrix,
  annot_raw,
  min_reads_per_cell = 1600,
  max_quantile_read_per_cell = 95,
  n_top_features = 40000,
  norm_type = "CPM",
  n_dims = 10,
  remove_PC = NULL,
  subsample_n = NULL,
  ref_genome = "hg38",
  exclude_regions = NULL,
  doBatchCorr = FALSE,
  batch_sels = NULL
)
```

Arguments

`datamatrix`	A sparse count matrix of features x cells.
`annot_raw`	A data.frame with barcode, cell_id, sample_id, batch_id, total_counts
`min_reads_per_cell`	Minimum read per cell to keep the cell
`max_quantile_read_per_cell`	Upper count quantile threshold above which cells are removed
`n_top_features`	Number of features to keep
`norm_type`	Normalization type c("CPM", "TFIDF", "RPKM", "TPM", "feature_size_only")

n_dims	An integer specifying the number of dimensions to keep for PCA
remove_PC	A vector of string indicating which principal components to remove before downstream analysis as probably correlated to library size. Should be under the form : 'Component_1', 'Component_2', ... Recommended when using 'TFIDF' normalization method. (NULL)
subsample_n	Number of cells to subsample.
ref_genome	Reference genome ("hg38" or "mm10").
exclude_regions	GenomicRanges with regions to remove from the object.
doBatchCorr	Run batch correction ? TRUE or FALSE
batch_sels	If doBatchCorr is TRUE, List of characters. Names are batch names, characters are sample names.

Value

A SingleCellExperiment object containing feature spaces.

Examples

```
raw <- create_scDataset_raw()
scExp = preprocessing_filtering_and_reduction(raw$mat, raw$annot)
```

---

preprocess_CPM *Preprocess scExp - Counts Per Million (CPM)*

Description

Preprocess scExp - Counts Per Million (CPM)

Usage

```
preprocess_CPM(scExp)
```

Arguments

scExp	A SingleCellExperiment Object

Value

A SingleCellExperiment object.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp = preprocess_CPM(scExp)
head(SingleCellExperiment::normcounts(scExp))
```

preprocess_feature_size_only *Preprocess scExp - size only*

Description

Preprocess scExp - size only

Usage

```
preprocess_feature_size_only(scExp)
```

Arguments

scExp	A SingleCellExperiment Object

Value

A SingleCellExperiment object.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp = preprocess_feature_size_only(scExp)
head(SingleCellExperiment::normcounts(scExp))
```

preprocess_RPKM *Preprocess scExp - Read per Kilobase Per Million (RPKM)*

Description

Preprocess scExp - Read per Kilobase Per Million (RPKM)

Usage

```
preprocess_RPKM(scExp)
```

Arguments

scExp	A SingleCellExperiment Object

Value

A SingleCellExperiment object.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp = preprocess_RPKM(scExp)
head(SingleCellExperiment::normcounts(scExp))
```

`preprocess_TFIDF` *Preprocess scExp - TF-IDF*

Description

Preprocess scExp - TF-IDF

Usage

```
preprocess_TFIDF(scExp, scale = 10000, log = TRUE)
```

Arguments

`scExp`	A SingleCellExperiment Object
`scale`	A numeric to multiply the matrix in order to have human readeable numbers. Has no impact on the downstream analysis
`log`	Wether to use neperian log on the TF-IDF normalized data or not.

Value

A SingleCellExperiment object.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp = preprocess_TFIDF(scExp)
head(SingleCellExperiment::normcounts(scExp))
```

`preprocess_TPM` *Preprocess scExp - Transcripts per Million (TPM)*

Description

Preprocess scExp - Transcripts per Million (TPM)

Usage

```
preprocess_TPM(scExp)
```

Arguments

`scExp`	A SingleCellExperiment Object

Value

A SingleCellExperiment object.

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp = preprocess_TPM(scExp)
head(SingleCellExperiment::normcounts(scExp))
```

`rawfile_ToBigWig` *rawfile_ToBigWig : reads in BAM file and write out BigWig coverage file, normalized and smoothed*

Description

rawfile_ToBigWig : reads in BAM file and write out BigWig coverage file, normalized and smoothed

Usage

```
rawfile_ToBigWig(
  input,
  BigWig_filename,
  format = "BAM",
  bin_width = 150,
  norm_factor,
  n_smoothBin = 5,
  ref = "hg38",
  read_size = 101,
```

```
  original_bins = NULL,
  quantile_for_peak_calling = 0.85
)
```

Arguments

input	Either a named list of character vector of path towards single-cell BED files or a sparse raw matrix of small bins («500bp). If a named list specifying scBEDn the names MUST correspond to the 'sample_id' column in your SingleCellExperiment object. The single-cell BED files names MUST match the barcode names in your SingleCellExperiment (column 'barcode'). The scBED files can be gzipped or not.
BigWig_filename	Path to write the output BigWig file
format	File format, either "BAM" or "BED"
bin_width	Bin size for coverage
norm_factor	Then number of cells or total number of reads in the given sample, for normalization.
n_smoothBin	Number of bins for smoothing values
ref	Reference genome.
read_size	Length of the reads.
original_bins	Original bins GenomicRanges in case the format is raw matrix.
quantile_for_peak_calling	The quantile to define the threshold above which signal is considered as a peak.

Value

Writes in the output directory a bigwig file displaying the cumulative coverage of cells and a basic set of peaks called by taking all peaks above a given threshold

Writes a BigWig file as output

---

raw_counts_to_sparse_matrix

Create a sparse count matrix from various format of input data.

Description

This function takes three different type of single-cell input: - Single cell BAM files (sorted) - Single cell BED files (gzipped) - A combination of an index file, a peak file and cell barcode file (The index file is composed of three column: index i, index j and value x for the non zeroes entries in the sparse matrix.)

Usage

```
raw_counts_to_sparse_matrix(
  files_dir_list,
  file_type = c("scBED", "scBAM", "FragmentFile"),
  use_Signac = TRUE,
  peak_file = NULL,
  n_bins = NULL,
  bin_width = NULL,
  genebody = NULL,
  extendPromoter = 2500,
  verbose = TRUE,
  ref = c("hg38", "mm10", "ce11")[1],
  progress = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

files_dir_list	A named character vector of directories containing the files. The names correspond to sample names.
file_type	Input file(s) type(s) ('scBED','scBAM','FragmentFile')
use_Signac	Use Signac wrapper function 'FeatureMatrix' if the Signac package is installed (TRUE).
peak_file	A file containing genomic location of peaks (NULL)
n_bins	The number of bins to tile the genome (NULL)
bin_width	The size of bins to tile the genome (NULL)
genebody	Count on genes (body + promoter) ? (NULL)
extendPromoter	If counting on genes, number of base pairs to extend up or downstream of TSS (2500).
verbose	Verbose (TRUE)
ref	reference genome to use (hg38)
progress	Progress object for Shiny
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Details

This functions re-counts signal on either fixed genomic bins, a set of user-defined peaks or around the TSS of genes.

Value

A sparse matrix of features x cells

References

Stuart el al., Multimodal single-cell chromatin analysis with Signac bioRxiv https://doi.org/10.1101/2020.11.09.373613

`read_count_mat_with_separated_chr_start_end` *Read a count matrix with three first columns (chr,start,end)*

Description

Read a count matrix with three first columns (chr,start,end)

Usage

```
read_count_mat_with_separated_chr_start_end(
  path_to_matrix,
  format_test,
  separator
)
```

Arguments

`path_to_matrix`	Path to the count matrix
`format_test`	Sample of the read.table
`separator`	Separator character

Value

A sparseMatrix with rownames in the form "chr1:1222-55555"

`read_sparse_matrix` *Read in one or multiple sparse matrices (10X format)*

Description

Given one or multiple directories, look in each directory for a combination of the following files :

- A 'features' file containing unique feature genomic locations -in tab separated format (*_features.bed / .txt / .tsv / .gz), e.g. chr, start and end
- A 'barcodes' file containing unique barcode names (_barcode.txt / .tsv / .gz)
- A 'matrix' A file containing indexes of non zero entries (_matrix.mtx / .gz)

Usage

```
read_sparse_matrix(
  files_dir_list,
  ref = c("hg38", "mm10", "ce11")[1],
  verbose = TRUE
)
```

Arguments

files_dir_list	A named character vector containing the full path towards folders. Each folder should contain only the Feature file, the Barcode file and the Matrix file (see description).
ref	Reference genome (used to filter non-canonical chromosomes).
verbose	Print ?

Value

Returns a list containing a datamatrix and cell annotation

Examples

```
## Not run:
sample_dirs = c("/path/to/folder1/", "/path/to/folder2/")
names(sample_dirs) = c("sample_1", "sample_2")
out <- read_sparse_matrix(sample_dirs, ref = "hg38")
head(out$datamatrix)
head(out$annot_raw)

## End(Not run)
```

---

rebin_helper *Rebin Helper for rebin_matrix function*

Description

Rebin Helper for rebin_matrix function

Usage

```
rebin_helper(mat_df)
```

Arguments

mat_df	A data.frame corresponding to sparse matrix indexes & values.

Value

a data.frame grouped mean-summarised by col and new_row

`rebin_matrix` *Transforms a bins x cells count matrix into a larger bins x cells count matrix.*

Description

This functions is best used to re-count large number of small bins or peaks (e.g. <= 5000bp) into equal or larger sized bins. The genome is either cut in fixed bins (e.g. 50,000bp) or into an user defined number of bins. Bins are calculated based on the canconical chromosomes. Note that if peaks are larger than bins, or if peaks are overlapping multiple bins, the signal is added to each bin. Users can increase the minimum overlap to consider peaks overlapping bins (by default 150bp, size of a nucleosome) to disminish the number of peaks overlapping multiple region. Any peak smaller than the minimum overlapp threshold will be dismissed. Therefore, library size might be slightly different from peaks to bins if signal was duplicated into multiple bins or ommitted due to peaks smaller than minimum overlap.

Usage

```
rebin_matrix(
  mat,
  bin_width = 50000,
  custom_annotation = NULL,
  minoverlap = 500,
  verbose = TRUE,
  ref = "hg38",
  nthreads = 1,
  rebin_function = rebin_helper
)
```

Arguments

`mat`	A matrix of peaks x cells
`bin_width`	Width of bins to produce in base pairs (minimum 500) (50000)
`custom_annotation`	A GenomicRanges object specifying the new features to count the matrix on instead of recounting on genomic bins. If not NULL, takes predecency over bin_width.
`minoverlap`	Minimum overlap between the original bins and the new features to consider the peak as overlapping the bin . We recommand to put this number at exactly half of the original bin size (e.g. 500bp for original bin size of 1000bp) so that no original bins are counted twice. (500)
`verbose`	Verbose
`ref`	Reference genome to use (hg38)
`nthreads`	Number of threads to use for paralell processing
`rebin_function`	A function to use to rebin the matrix.

Value

A sparse matrix of larger bins or peaks.

Examples

```
mat = create_scDataset_raw()$mat
binned_mat = rebin_matrix(mat,bin_width = 10e6)
dim(binned_mat)
```

---

reduce_dims_scExp *Reduce dimensions (PCA, TSNE, UMAP)*

Description

Reduce dimensions (PCA, TSNE, UMAP)

Usage

```
reduce_dims_scExp(
  scExp,
  dimension_reductions = c("PCA", "UMAP"),
  n = 10,
  batch_correction = FALSE,
  batch_list = NULL,
  remove_PC = NULL,
  verbose = TRUE
)
```

Arguments

scExp	A SingleCellExperiment object.
dimension_reductions	A character vector of methods to apply. (c('PCA','TSNE','UMAP'))
n	Numbers of dimensions to keep for PCA. (50)
batch_correction	Do batch correction ? (FALSE)
batch_list	List of characters. Names are batch names, characters are sample names.
remove_PC	A vector of string indicating which principal components to remove before downstream analysis as probably correlated to library size. Should be under the form : 'Component_1', 'Component_2', ... Recommended when using 'TFIDF' normalization method. (NULL)
verbose	Print messages ?(TRUE)

Value

A SingleCellExperiment object containing feature spaces. See ?reduceDims().

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp =  normalize_scExp(scExp, "CPM")
scExp = reduce_dims_scExp(scExp,dimension_reductions=c("PCA","UMAP"))
```

---

reduce_dim_batch_correction *Reduce dimension with batch corrections*

Description

Reduce dimension with batch corrections

Usage

```
reduce_dim_batch_correction(scExp, mat, batch_list, n)
```

Arguments

scExp	SingleCellExperiment
mat	The normalized count matrix
batch_list	List of batches
n	Number of PCs to keep

Value

A list containing the SingleCellExperiment with batch info and the corrected pca

---

remove_chr_M_fun *Remove chromosome M from scExprownames*

Description

Remove chromosome M from scExprownames

Usage

```
remove_chr_M_fun(scExp, verbose)
```

Arguments

scExp	A SingleCellExperiment
verbose	Print ?

Value

A SingleCellExperiment without chromosome M (mitochondrial chr)

`remove_non_canonical_fun`
Remove non canonical chromosomes from scExp

Description

Remove non canonical chromosomes from scExp

Usage

```
remove_non_canonical_fun(scExp, verbose)
```

Arguments

`scExp`	A SingleCellExperiment
`verbose`	Print ?

Value

A SingleCellExperiment without non canonical chromosomes (random,unknown, contigs etc...)

`results_enrichmentTest`
Resutls of hypergeometric gene set enrichment test

Description

Run hypergeometric enrichment test and combine significant pathways into a data.frame

Usage

```
results_enrichmentTest(
  differentialGenes,
  enrichment_qval,
  GeneSets,
  GeneSetsDf,
  GenePool
)
```

Arguments

differentialGenes	Genes significantly over / under expressed
enrichment_qval	Adusted p-value threshold above which a pathway is considered significative
GeneSets	List of pathways
GeneSetsDf	Data.frame of pathways
GenePool	Pool of possible genes for testing

Value

A data.frame with pathways passing q.value threshold

---

retrieve_top_bot_features_pca

Retrieve Top and Bot most contributing features of PCA

Description

Retrieve Top and Bot most contributing features of PCA

Usage

```
retrieve_top_bot_features_pca(
  pca,
  counts,
  component,
  n_top_bot,
  absolute = FALSE
)
```

Arguments

pca	A matrix/data.frame of rotated data
counts	the normalized counts used for PCA
component	the componenent of interest
n_top_bot	the number of top & bot features to take
absolute	If TRUE, return the top features in absolute values instead.

Value

a data.frame of top bot contributing features in PCA

run_pairwise_tests *Run pairwise tests*

Description

Run pairwise tests

Usage

```
run_pairwise_tests(
  affectation,
  by,
  counts,
  feature,
  method,
  progress = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

affectation	An annotation data.frame with cell_cluster and cell_id columns
by	= A character specifying the column of the object containing the groups of cells to compare.
counts	Count matrix
feature	Feature data.frame
method	DA method, Wilcoxon or edgeR
progress	A shiny Progress instance to display progress bar.
BPPARAM	BPPARAM object for multiprocessing. See bpparam for more informations. Will take the default BPPARAM set in your R session.

Value

A list containing objects for DA function

run_tsne_scExp *Run tsne on single cell experiment*

Description

Run tsne on single cell experiment

Usage

```
run_tsne_scExp(scExp, verbose = FALSE)
```

Arguments

scExp	A SingleCellExperiment Object
verbose	Print ?

Value

A colored kable with the number of cells per sample for display

---

scExp *A SingleCellExperiment outputed by ChromSCape*

Description

Data from a single-cell ChIP-seq experiment against H3K4me3 active mark from two cell lines, Jurkat B cells and Ramos T cells from Grosselin et al., 2019. The count matrices, on 5kbp bins, were given to ChromSCape and the filtering parameter was set to 3% of cells active in regions and subsampled down to 150 cells per sample. After correlation filtering, the experiment is composed of respectively 51 and 55 cells from Jurkat & Ramos and 5499 5kbp-genomic bins where signal is located.

Usage

```
data("scExp")
```

Format

scExp - a SingleCellExperiment with 106 cells and 5499 features (genomic bins) in hg38:

chr A SingleCellExperiment

Details

The scExp is composed of :

- counts and normcounts assays, PCA, UMAP, and Correlation matrix in reducedDims(scExp)
- Assignation of genes to genomic bins in rowRanges(scExp)
- Cluster information in colData(scExp) correlation
- Hierarchical clustering dengogram in metadata$hc_cor
- Consensus clustering raw data in metadata$consclust
- Consensus clustering cluster-consensus and item consensus dataframes in metadata$icl
- Differential analysis in metadata$diff
- Gene Set Analysis in metadata$enr

Value

A scExp - a SingleCellExperiment with 106 cells and 5499 features

Examples

```
data("scExp")
plot_reduced_dim_scExp(scExp)
plot_reduced_dim_scExp(scExp,color_by = "cell_cluster")
plot_heatmap_scExp(scExp)
plot_differential_volcano_scExp(scExp,  "C1")
plot_differential_summary_scExp(scExp)
```

`separate_BAM_into_clusters`

Separate BAM files into cell cluster BAM files

Description

Separate BAM files into cell cluster BAM files

Usage

```
separate_BAM_into_clusters(affectation, odir, merged_bam)
```

Arguments

`affectation`	An annotation data.frame containing cell_id and cell_cluster columns
`odir`	A valid output directory path
`merged_bam`	A list of merged bam file paths
	@importFrom Rsamtools filterBam ScanBamParam

Value

Create one BAM per cluster from one BAM per condition

separator_count_mat *Determine Count matrix separator ("tab" or ",")*

Description

Determine Count matrix separator ("tab" or ",")

Usage

```
separator_count_mat(path_to_matrix)
```

Arguments

path_to_matrix	A path towards the count matrix to check

Value

A character separator

smoothBin *Smooth a vector of values with nb_bins left and righ values*

Description

Smooth a vector of values with nb_bins left and righ values

Usage

```
smoothBin(bin_score, nb_bins = 10)
```

Arguments

bin_score	A numeric vector of values to be smoothed
nb_bins	Number of values to take left and right @importFrom BiocParallel bpvec

Value

A smooth vector of the same size

subsample_scExp *Subsample scExp*

Description

Randomly sample x cells from each sample in a SingleCellExperiment to return a subsampled SingleCellExperiment with all samples having maximum n cells. If n is higher than the number of cell in a sample, this sample will not be subsampled.

Usage

```
subsample_scExp(scExp, n_cell_per_sample = 500, n_cell_total = NULL)
```

Arguments

scExp	A SingleCellExperiment
n_cell_per_sample	An integer number of cells to subsample for each sample. Exclusive with n_cells_total. (500)
n_cell_total	An integer number of cells to subsample in total. Exclusive with n_cell_per_sample (NULL).

Value

A subsampled SingleCellExperiment

Examples

```
raw <- create_scDataset_raw()
scExp = create_scExp(raw$mat, raw$annot)
scExp_sub = subsample_scExp(scExp,50)
## Not run: num_cell_scExp(scExp_sub)
```

subset_bam_call_peaks *Peak calling on cell clusters*

Description

This functions does peak calling on each cell population in order to refine gene annotation for large bins. For instance, a 50000bp bins might contain the TSS of several genes, while in reality only one or two of these genes are overlapping the signal (peak). To do so, first in-silico cell sorting is applied based on previously defined clusters contained in the SingleCellExperiment. Taking BAM files of each sample as input, samtools pools then splits reads from each cell barcode into 1 BAM file per cell cluster (pseudo-bulk). Then MACS2 calls peaks on each cluster. The peaks are aggregated and merged if closer to a certain distance defined by user (10000bp). Then,

This function takes as input a SingleCellExperiment, that must contain a 'cell_cluster' column in it's colData, an output directory where to store temporary files, the list of BAM files corresponding to each sample and containing the cell barcode information as a tag (for instance tag CB:Z:xxx, XB:Z:xxx or else...) or single-cell BED files containing the raw reads and corresponding to the 'barcode' column metadata, the p.value used by MACS2 to distinguish significant peaks, the reference genome (either hg38 or mm10), the maximal merging distance in bp and a data.frame containing gene TSS genomic cooridnates of corresponding genome (if set to NULL, will automatically load geneTSS). The output is a SingleCellExperiment with GRanges object containing ranges of each merged peaks that falls within genomic bins of the SingleCellExperiment, saving the bin range as additional column (window_chr, window_start, window_end), as well as the closests genes and their distance relative to the peak. The peaks may be present in several rows if multiple genes are close / overlap to the peaks.

Note that the user must have MACS2 installed and available in the PATH. Users can open command terminal and type 'which macs2' to verify the availability of these programs. Will only work on unix operating system. Check operating system with 'print(.Platform)'.

Usage

```
subset_bam_call_peaks(
  scExp,
  odir,
  input,
  format = "BAM",
  p.value = 0.05,
  ref = "hg38",
  peak_distance_to_merge = 10000,
  geneTSS_annotation = NULL,
  run_coverage = FALSE,
  progress = NULL
)
```

Arguments

scExp	A SingleCellExperiment object
odir	Output directory where to write temporary files and each cluster's BAM file
input	A character vector of file paths to each sample's BAM file, containing cell barcode information as tags. BAM files can be paired-end or single-end.
format	Format of the input data, either "BAM" or "scBED".
p.value	a p-value to use for MACS2 to determine significant peaks. (0.05)
ref	A reference genome, either hg38, mm10 or ce11. ('hg38')
peak_distance_to_merge	Maximal distance to merge peaks together after peak calling , in bp. (10000)
geneTSS_annotation	A data.frame annotation of genes TSS. If NULL will automatically load Gencode list of genes fro specified reference genome.
run_coverage	Create coverage tracks (.bw) for each cluster ?
progress	A shiny Progress instance to display progress bar.

Details

The BED files of the peaks called for each clusters, as well as the merged peaks are written in the output directory.

Value

A SingleCellExperiment with refinded annotation

Examples

```
## Not run:
data("scExp")
subset_bam_call_peaks(scExp, "path/to/out/", list("sample1" =
   "path/to/BAM/sample1.bam", "sample2" = "path/to/BAM/sample2.bam"),
   p.value = 0.05, ref = "hg38", peak_distance_to_merge = 10000,
   geneTSS_annotation = NULL)

## End(Not run)
```

---

summary_DA *Summary of the differential analysis*

Description

Summary of the differential analysis

Usage

```
summary_DA(scExp, qval.th = 0.01, logFC.th = 1, min.percent = 0.01)
```

Arguments

scExp	A SingleCellExperiment object containing consclust with selected number of cluster.
qval.th	Adjusted p-value threshold. (0.01)
logFC.th	Fold change threshold. (1)
min.percent	Minimum fraction of cells having the feature active to consider it as significantly differential. (0.01)

Value

A table summary of the differential analysis

Examples

```
data('scExp')
summary_DA(scExp)
```

swapAltExp_sameColData

Swap main & alternative Experiments, with fixed colData

Description

Swap main & alternative Experiments, with fixed colData

Usage

```
swapAltExp_sameColData(scExp, alt)
```

Arguments

scExp	A SingleCellExperiment
alt	Name of the alternative experiment

Value

A swapped SingleCellExperiment with the exact same colData.

Examples

```
data(scExp)
swapAltExp_sameColData(scExp, "peaks")
```

table_enriched_genes_scExp

Creates table of enriched genes sets

Description

Creates table of enriched genes sets

Usage

```
table_enriched_genes_scExp(
  scExp,
  set = "Both",
  group = "C1",
 enr_class_sel = c("c1_positional", "c2_curated", "c3_motif", "c4_computational",
    "c5_GO", "c6_oncogenic", "c7_immunologic", "hallmark")
)
```

Arguments

scExp	A SingleCellExperiment object containing list of enriched gene sets.
set	A character vector, either 'Both', 'Overexpressed' or 'Underexpressed'. ('Both')
group	The "group" name from differential analysis. Can be the cluster name or the custom name in case of a custom differential analysis.
enr_class_sel	Which classes of gene sets to show. (c('c1_positional', 'c2_curated', ...))

Value

A DT::data.table of enriched gene sets.

Examples

```
data("scExp")
## Not run: table_enriched_genes_scExp(scExp)
```

---

warning_DA *Warning for differential_analysis_scExp*

Description

Warning for differential_analysis_scExp

Usage

```
warning_DA(scExp, by, de_type, method, block, group, ref)
```

Arguments

scExp	A SingleCellExperiment object containing consclust with selected number of cluster.
by	= A character specifying the column of the object containing the groups of cells to compare. Exclusive with de_type == custom
de_type	Type of comparisons. Either 'one_vs_rest', to compare each cluster against all others, or 'pairwise' to make 1 to 1 comparisons. ('one_vs_rest')
method	Wilcoxon or edgerGLM
block	Use batches as blocking factors ?
group	If de_type is custom, the group to compare (data.frame), must be a one-column data.frame with cell_clusters or sample_id as character in rows
ref	If de_type is custom, the reference to compare (data.frame), must be a one-column data.frame with cell_clusters or sample_id as character in rows

Value

Warnings or Errors if the input are not correct

warning_filter_correlated_cell_scExp
warning_filter_correlated_cell_scExp

Description

warning_filter_correlated_cell_scExp

Usage

```
warning_filter_correlated_cell_scExp(
  scExp,
  random_iter,
  corr_threshold,
  percent_correlation,
  run_tsne,
  downsample,
  verbose
)
```

Arguments

scExp	A SingleCellExperiment object containing 'Cor', a correlation matrix, in reducedDims.
random_iter	Number of random matrices to create to calculate random correlation scores. (50)
corr_threshold	Quantile of random correlation score above which a cell is considered to be 'correlated' with another cell. (99)
percent_correlation	Percentage of the cells that any cell must be 'correlated' to in order to not be filtered. (1)
run_tsne	Re-run tsne ? (FALSE)
downsample	Number of cells to calculate correlation filtering threshold ? (2500)
verbose	(TRUE)

Value

Warnings or Errors if the input are not correct

`warning_plot_reduced_dim_scExp` *A warning helper for plot_reduced_dim_scExp*

Description

A warning helper for plot_reduced_dim_scExp

Usage

```
warning_plot_reduced_dim_scExp(
  scExp,
  color_by,
  reduced_dim,
  downsample,
  transparency,
  size,
  max_distanceToTSS,
  annotate_clusters,
  min_quantile,
  max_quantile
)
```

Arguments

`scExp`	A SingleCellExperiment Object
`color_by`	Feature used for coloration
`reduced_dim`	Reduced Dimension used for plotting
`downsample`	Number of cells to downsample
`transparency`	Alpha parameter, between 0 and 1
`size`	Size of the points.
`max_distanceToTSS`	Numeric. Maximum distance to a gene's TSS to consider a region linked to a gene.
`annotate_clusters`	A logical indicating if clusters should be labelled. The 'cell_cluster' column should be present in metadata.
`min_quantile`	The lower threshold to remove outlier cells, as quantile of cell embeddings (between 0 and 0.5).
`max_quantile`	The upper threshold to remove outlier cells, as quantile of cell embeddings (between 0.5 and 1).

Value

Warning or errors if the inputs are not correct

warning_raw_counts_to_sparse_matrix *Warning for raw_counts_to_sparse_matrix*

Description

Warning for raw_counts_to_sparse_matrix

Usage

```
warning_raw_counts_to_sparse_matrix(
  files_dir_list,
  file_type = c("scBAM", "scBED", "SparseMatrix"),
  peak_file = NULL,
  n_bins = NULL,
  bin_width = NULL,
  genebody = NULL,
  extendPromoter = 2500,
  verbose = TRUE,
  ref = "hg38"
)
```

Arguments

files_dir_list	A named character vector of directory containing the raw files
file_type	Input file(s) type(s) ('scBED','scBAM','SparseMatrix')
peak_file	A file containing genomic location of peaks (NULL)
n_bins	The number of bins to tile the genome (NULL)
bin_width	The size of bins to tile the genome (NULL)
genebody	Count on genes (body + promoter) ? (NULL)
extendPromoter	If counting on genes, number of base pairs to extend up or downstream of TSS (2500).
verbose	Verbose (TRUE)
ref	reference genome to use (hg38)

Value

Error or warnings if the input are not correct

`wrapper_Signac_FeatureMatrix` *Wrapper around 'FeatureMatrix' function from Signac Package*

Description

Wrapper around 'FeatureMatrix' function from Signac Package

Usage

```
wrapper_Signac_FeatureMatrix(
  files_dir_list,
  which,
  ref = "hg38",
  process_n = 2000,
  set_future_plan = TRUE,
  verbose = TRUE,
  progress = NULL
)
```

Arguments

`files_dir_list`	A named character vector of directories containing the files. The names correspond to sample names.
`which`	A GenomicRanges containing the features to count on.
`ref`	Reference genome to use (hg38).Chromosomes that are not present in the canonical chromosomes of the given reference genome will be excluded from the matrix.
`process_n`	Number of regions to load into memory at a time, per thread. Processing more regions at once can be faster but uses more memory. (2000)
`set_future_plan`	Set 'multisession' plan within the function (TRUE). If TRUE, the previous plan (e.g. future::plan()) will be set back on exit.
`verbose`	Verbose (TRUE).
`progress`	Progress object for Shiny.

Details

Signac & future are not required packages for ChromSCape as they are required only for the fragment matrix calculations. To use this function, install Signac package first (future will be installed as a dependency). For the simplicity of the application & optimization, the function by defaults sets future::plan("multisession") with workers = future::availableCores(omit = 1) in order to allow parallel processing with Signac. On exit the plan is re-set to the previously set future plan. Note that future multisession may have trouble running when VPN is on. To run in parallel, first deactivate your VPN if you encounter long runtimes.

Value

A sparse matrix of features x cells

References

Stuart el al., Multimodal single-cell chromatin analysis with Signac bioRxiv https://doi.org/10.1101/2020.11.09.373613

Examples

```
## Not run:
gr_bins = define_feature("hg38", bin_width = 50000)
wrapper_Signac_FeatureMatrix("/path/to/dir_containing_fragment_files",
 gr_bins, ref = "hg38")

## End(Not run)
```

Index