Package ‘MOCHA’

January 25, 2024

Type Package

Title Modeling for Single-Cell Open Chromatin Analysis

Version 1.1.0

Maintainer Imran McGrath <aifi.combio.support@alleninstitute.org>

Description A statistical framework and analysis tool for open chromatin analysis designed specifically for single cell ATAC-seq (Assay for Transposase-Accessible Chromatin) data, after cell type/cluster identification. These novel modules remove unwanted technical variation, identify open chromatin, robustly models repeated measures in single cell data, implement advanced statistical frameworks to model zero-inflation for differential and co-accessibility analyses, and integrate with existing databases and modules for downstream analyses to reveal biological insights. MOCHA provides a statistical foundation for complex downstream analysis to help advance the potential of single cell ATAC-seq for applied studies. Methods for zero-inflated statistics are as described in: Ghazanfar, S., Lin, Y., Su, X. et al. (2020) <doi:10.1038/s41592-020-0885-x>. Pimentel, Ronald Silva, ``Kendall’s Tau and Spearman’s Rho for Zero-Inflated Data” (2009) <https://scholarworks.wmich.edu/dissertations/721/>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Depends R (>= 4.1.0)

biocViews

Imports data.table, plyranges (>= 1.14.0), dplyr, GenomicRanges, RaggedExperiment, MultiAssayExperiment, SummarizedExperiment, stringr, ggbio, wCorr, magrittr, rlang, AnnotationDbi, BiocGenerics, GenomeInfoDb, GenomicFeatures, IRanges, S4Vectors, assertthat, ensembldb, ggplot2, ggrepel, matrixStats, methods, qvalue, scales, tidyr, ggridges, pbapply, BSGenome, tidyselect, lifecycle

1
Suggests ArchR, RMariaDB, motifmatchr, BiocManager,
TxDb.Hsapiens.UCSC.hg38.refGene,
TxDb.Hsapiens.UCSC.hg19.knownGene, org.Hs.eg.db,
BSgenome.Hsapiens.UCSC.hg19, withr, knitr, rmarkdown, chromVAR,
testthat (>= 3.0.0), uwot, irlba, glmmTMB, Matrix, waldo,
purrr, lmerTest, Biobase, lme4, zip, rtracklayer, cowplot,
mixtools, zoo

Additional_repositories https://imran-aifi.github.io/drat

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Samir Rachid Zaim [aut, ctb],
Mark-Phillip Pebworth [aut, ctb],
Imran McGrath [aut, cre],
Lauren Okada [aut, ctb],
Xiaojun Li [aut, ctb]

Repository CRAN

Date/Publication 2024-01-25 12:20:12 UTC

R topics documented:

.. counts_plot_default_theme ........................................ 3
.. gene_plot_theme ................................................... 4
addAccessibilityShift ................................................ 4
addMotifSet ........................................................... 5
annotateTiles ............................................................ 6
bulkDimReduction ....................................................... 7
bulkUMAP ................................................................. 8
callOpenTiles ........................................................... 9
combineSampleTileMatrix .............................................. 12
differentialsToGRanges .............................................. 13
exampleBlackList ...................................................... 13
exampleCellColData .................................................... 14
exampleFragments ...................................................... 14
exportCoverage .......................................................... 15
exportDifferentials .................................................. 16
exportMotifs ............................................................ 17
exportOpenTiles ......................................................... 18
exportSmoothedInsertions ........................................... 19
extractRegion .......................................................... 20
filterCoAccessibleLinks .............................................. 21
finalModelObject ........................................................ 22
getAltTSS ............................................................... 23
getAnnotationDbFromInstalledPkgname ................................ 24
getCellPopMatrix ...................................................... 24
Default ggplot theme for counts plot

Description

Default ggplot theme for counts plot

Usage

.counts_plot_default_theme
addAccessibilityShift

**Format**
An object of class list of length 10.

```
.addAccessibilityShift
```

**Description**
Common theme for gene plots

**Usage**
.addAccessibilityShift

```
.addAccessibilityShift
```

**Description**
addAccessibilityShift will add a new condition to the SummarizedExperiment output of extractRegion, which will contain the difference in accessibility between two conditions

**Usage**
addAccessibilityShift(CountSE, foreground, background, assayName = NULL)

**Arguments**
- CountSE: The SummarizedExperiment object output from extractRegion
- foreground: Group that will be used as the foreground for the subtraction of accessibility
- background: Group that will be used as the background for the subtraction of accessibility
- assayName: The name given to the new assay that is difference in accessibility between foreground and background.

**Value**
countSE a SummarizedExperiment containing coverage for the given input cell populations.
addMotifSet

Examples

## Not run:
# CountSE is a SummarizedExperiment generated by extractRegion()
countSE <- MOCHA::addAccessibilityShift(
  CountSE = CountSE,
  foreground = "Condition1",
  background = "Condition2",
  assayName = "AccessibilityChanges"
)

## End(Not run)

addMotifSet addMotifSet

Description

addMotifSet Identify motifs within your peakset.

Usage

addMotifSet(
  SampleTileObj,
  motifPWMs,
  w = 7,
  returnSTM = TRUE,
  motifSetName = "Motifs"
)

Arguments

SampleTileObj A SummarizedExperiment, specifically the output of getSampleTileMatrix
motifPWMs A pwms object for the motif database. Either PFMatrix, PFMatrixList, PWMa-
trix, or PWMatrixList
w Parameter for motifmatchr controlling size in basepairs of window for filtration. Default is 7.
returnSTM If TRUE, return the modified SampleTileObj with motif set added to metadata (default). If FALSE, return the motifs from motifmatchr as a GRanges.
motifSetName Name to give motifList in the SampleTileObj’s metadata if ‘returnSTM=TRUE’. Default is ‘Motifs’.

Value

the modified SampleTileObj with motifs added to the metadata
## Not run:
```
# load a curated motif set from library(chromVARmotifs)
# included with ArchR installation
data(human_pwms_v2)
SE_with_motifs <- addMotifSet(
  SampleTileObj,
  motifPWMs = human_pwms_v2,
  returnSTM = TRUE, motifSetName = "Motifs", w = 7
)
```
## End(Not run)

### Examples

```r
tile_matrix <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 3, ncol = 2)
rownames(tile_matrix) <- c("A", "B", "C")
colnames(tile_matrix) <- c("X", "Y")
```

### Description

annotateTiles annotates a set of sample-tile matrices given with gene annotations. Details on TxDb and Org annotation packages and available annotations can be found at Bioconductor: https://bioconductor.org/packages/

### Usage

```r
annotateTiles(Obj, TxDb = NULL, Org = NULL, promoterRegion = c(2000, 100))
```

### Arguments

- **Obj**: A RangedSummarizedExperiment generated from getSampleTileMatrix, containing TxDb and Org in the metadata. This may also be a GRanges object.
- **TxDb**: The annotation package for TxDb object for your genome. Optional, only required if Obj is a GRanges.
- **Org**: The genome-wide annotation for your organism. Optional, only required if Obj is a GRanges.
- **promoterRegion**: Optional list containing the window size in basepairs defining the promoter region. The format is (upstream, downstream). Default is (2000, 100).

### Value

Obj, the input data structure with added gene annotations (whether GRanges or SampleTileObj)

### Examples

```r
## Not run:
library(TxDb.Hsapiens.UCSC.hg38.refGene)
library(org.Hs.eg.db)
SampleTileMatricesAnnotated <- MOCHA::annotateTiles(
  SampleTileMatrices,
  TxDb = NULL, Org = NULL, promoterRegion = c(2000, 100)))
```
bulkDimReduction runs dimensionality reduction (either PCA or LSI). We adapt Signac’s

Usage

bulkDimReduction(
  SampleTileObj,
  cellType = "All",
  componentNumber = 30,
  method = "LSI",
  verbose = FALSE
)

Arguments

SampleTileObj The SummarizedExperiment object output from getSampleTileMatrix

cellType vector of strings. Cell subsets for which to call peaks. This list of group names must be identical to names that appear in the SampleTileObj. Optional, if cellPopulations=’ALL’, then peak calling is done on all cell populations. Default is ’ALL’.

componentNumber integer. Number of components to include in LSI, or PCA. This must be strictly less than

method a string. Represents the method to use. Includes LSI or PCA, but we do not recommend PCA for scATAC pseudobulk.

verbose Set TRUE to display additional messages. Default is FALSE.

Value

SEObj a SummarizedExperiment containing PC components from dimensionality reduction and metadata from the SampleTileObj

References

LSI method adapted from Andrew Hill: http://andrewjohnhill.com/blog/2019/05/06/dimensionality-reduction-for-scatac-data/
bulkUMAP

Examples

```r
## Not run:
LSIObj <- MOCHA::bulkDimReduction(SampleTileObj, cellType = "CD16_Mono")
## End(Not run)
```

## Description

bulkUMAP generates UMAP from pseudobulk LSIObj object, and merges in metadata.

## Usage

```r
bulkUMAP(
  SEObj,
  assay = "LSI",
  components = c(1:30),
  nNeighbors = 15,
  returnModel = FALSE,
  seed = 1,
  ...
)
```

## Arguments

- **SEObj**: The SummarizedExperiment object output from bulkDimReduction, or an STM, subsetted down to just one cell type.
- **assay**: A string, describing the name of the assay within SEObj to run UMAP ('PCA', 'LSI', or 'counts').
- **components**: A vector of integers. Number of components to include in LSI (1:30 typically).
- **nNeighbors**: See umap. The size of local neighborhood (in terms of number of neighboring sample points) used for manifold approximation. Default is 15.
- **returnModel**: A boolean. Default is FALSE. If set to true, it will return a list, where the first is the UMAP coordinates with metadata for plotting, and the second is the full UMAP model so further projection can occur.
- **seed**: an integer. Represents the random seed to pass to the UMAP. Default seed is 1.
- **...**: Additional arguments to be passed to umap.

## Value

fullUMAP data.frame of UMAP values with metadata attached.
callOpenTiles

Examples

## Not run:
UMAPvalues <- MOCHA::bulkUMAP(LSIObj)

## End(Not run)

callOpenTiles Perform peak-calling on a set of fragments or an ArchR Project.

description

callOpenTiles is the main peak-calling function in MOCHA that serves as a wrapper function to call peaks provided a set of fragment files and an ArchR Project for meta-data purposes

Usage

callOpenTiles(
  ATACFragments,
  cellColData,
  blackList,
  genome,
  cellPopLabel,
  cellPopulations = "ALL",
  studySignal = NULL,
  generalizeStudySignal = FALSE,
  cellCol = "RG",
  TxDb,
  OrgDb,
  outDir,
  numCores = 30,
  verbose = FALSE,
  force = FALSE
)

## S4 method for signature 'GRangesList'
callOpenTiles(
  ATACFragments,
  cellColData,
  blackList,
  genome,
  cellPopLabel,
  cellPopulations = "ALL",
  studySignal = NULL,
  generalizeStudySignal = FALSE,
  cellCol = "RG",
  TxDb,
arg

## Arguments

ATACFragments an ArchR Project, or a GRangesList of fragments. Each GRanges in the GRanges list must have unique cell IDs in the column given by 'cellCol'.

cellColData A DataFrame containing cell-level metadata. This must contain both a column 'Sample' with unique sample IDs and the column specified by 'cellPopLabel'.

blackList A GRanges of blacklisted regions

gene A BSgenome object, or the full name of an installed BSgenome data package,
or a short string specifying the name of an NCBI assembly (e.g. "GRCh38", "TAIR10.1", etc...) or UCSC genome (e.g. "hg38", "bosTau9", "galGal6", "ce11", etc...). The supplied short string must refer unambiguously to an installed BSgenome data package. See getBSgenome.

cellPopLabel string indicating which column in the ArchRProject metadata contains the cell population label.

cellPopulations vector of strings. Cell subsets for which to call peaks. This list of group names must be identical to names that appear in the ArchRProject metadata. Optional, if cellPopulations='ALL', then peak calling is done on all cell populations in the ArchR project metadata. Default is 'ALL'.

studySignal The median signal (number of fragments) in your study. If not set, this will be calculated using the input ArchR project but relies on the assumption that the ArchR project encompasses your whole study (i.e. is not a subset).

generalizeStudySignal If `studySignal` is not provided, calculate the signal as the mean of the mean & median number of fragments for of individual samples within each cell population. This may improve MOCHA’s ability to generalize to datasets with XXXXX #TODO. Default is FALSE, use the median number of fragments.

cellCol The column in cellColData specifying unique cell ids or barcodes. Default is "RG", the unique cell identifier used by ArchR.

TxDb The exact package name of a TxDb-class transcript annotation package for your organism (e.g. "TxDb.Hsapiens.UCSC.hg38.refGene"). This must be installed. See Bioconductor AnnotationData Packages.

OrgDb The exact package name of a OrgDb-class genome wide annotation package for your organism (e.g. "org.Hs.eg.db"). This must be installed. See Bioconductor AnnotationData Packages

outDir is a string describing the output directory for coverage files. Must be a complete directory string. With ArchR input, set outDir to NULL to create a directory within the input ArchR project directory named MOCHA for saving files.

numCores integer. Number of cores to parallelize peak-calling across multiple cell populations.

verbose Set TRUE to display additional messages. Default is FALSE.

force Optional, whether to force creation of coverage files if they already exist. Default is FALSE.

Value

tileResults A MultiAssayExperiment object containing ranged data for each tile

Examples

```r
## Not run:
# Starting from an ArchR Project:
tileResults <- MOCHA::callOpenTiles(
  ArchRProj = myArchRProj,
)```
combineSampleTileMatrix

Description

combineSampleTileMatrix combines all celltypes in a SampleTileMatrix object into a SummarizedExperiment with one single matrix across all cell types and samples.

Usage

combineSampleTileMatrix(SampleTileObj, NAtoZero = TRUE, verbose = FALSE)

Arguments

SampleTileObj  The SummarizedExperiment object output from getSampleTileMatrix containing your sample-tile matrices
NAtoZero       Set NA values in the sample-tile matrix to zero
verbose        Set TRUE to display additional messages. Default is FALSE.
differentialsToGRanges

Value

TileCorr A data.table correlation matrix

differentialsToGRanges Converts a data.frame matrix to a GRanges, preserving additional columns as GRanges metadata

Description
differentialsToGRanges Converts a data.frame matrix to a GRanges, preserving additional columns as GRanges metadata

Usage
differentialsToGRanges(differentials, tileColumn = "Tile")

Arguments
differentials a matrix/data.frame with a column tileColumn containing region strings in the format "chr:start-end"
tileColumn name of column containing region strings. Default is "Tile".

Value

a GRanges containing all original information

exampleBlackList

Description

Example input of a blackList extracted from the PBMC_Small dataset consisting of 2k cells and spanning chr1 and 2 (~2-300MB). The data is publicly available with the ArchR package at <https://www.archrproject.com/reference/getTestProject.html>

Usage
eampleBlackList

Format

A GRanges object with 210 ranges and 2 metadata columns
Description

Example input of cellColData extracted from the PBMC_Small dataset consisting of 2k cells and spanning chr1 and 2 (~2-300MB). The data is publicly available with the ArchR package at <https://www.archrproject.com/reference/getTestProject.html>

Usage

exampleCellColData

Format

A DataFrame with 2217 rows and 3 columns

Description

Example input of ATAC fragments extracted from the PBMC_Small dataset consisting of 2k cells and spanning chr1 and 2 (~2-300MB). This subset consists of two cell populations: Clusters C2 and C5. The data is publicly available with the ArchR package at <https://www.archrproject.com/reference/getTestProject.html>

Usage

exampleFragments

Format

A list of 2 GRanges objects
Description

`exportCoverage` will export normalized coverage files to BigWig files, either as sample-specific or sample-averaged files, for visualization in genome browsers.

Usage

```r
exportCoverage(
  SampleTileObject,
  dir = getwd(),
  type = TRUE,
  cellPopulations = "ALL",
  groupColumn = NULL,
  subGroups = NULL,
  sampleSpecific = FALSE,
  saveFile = TRUE,
  numCores = 1,
  verbose = FALSE
)
```

Arguments

- **SampleTileObject**
  The SummarizedExperiment object output from `getSampleTileMatrix`

- **dir**
  string. Directory to save files to.

- **type**
  Boolean. Default is TRUE, and exports Coverage. If set to FALSE, exports Insertions.

- **cellPopulations**
  vector of strings. Cell subsets for which to call peaks. This list of group names must be identical to names that appear in the SampleTileObject. Optional, if cellPopulations='ALL', then peak calling is done on all cell populations. Default is 'ALL'.

- **groupColumn**
  Optional, the column containing sample group labels for returning coverage within sample groups. Default is NULL, all samples will be used.

- **subGroups**
  a list of subgroup(s) within the groupColumn from the metadata. Optional, default is NULL, all labels within groupColumn will be used.

- **sampleSpecific**
  If TRUE, a BigWig will export for each sample-cell type combination.

- **saveFile**
  Boolean. If TRUE, it will save to a BigWig. If FALSE, it will return the GRangesList without writing a BigWig.

- **numCores**
  integer. Number of cores to parallelize peak-calling across multiple cell populations

- **verbose**
  Set TRUE to display additional messages. Default is FALSE.
exportDifferentials

**Value**

countSE a SummarizedExperiment containing coverage for the given input cell populations.

**Examples**

```r
## Not run:
MOCHA::exportCoverage(
  SampleTileObject = SampleTileMatrices,
  cellPopulations = "ALL",
  numCores = 30,
  sampleSpecific = FALSE
)
## End(Not run)
```

---

**Description**

exportDifferentials exports the differential peaks output GRangesList output from getDifferentialAccessibleTiles to bigBed format for visualization in genome browsers.

**Usage**

```r
exportDifferentials(
  SampleTileObject, 
  DifferentialsGRList, 
  outDir, 
  verbose = FALSE 
)
```

**Arguments**

- **SampleTileObject**
  The SummarizedExperiment object output from getSampleTileMatrix
- **DifferentialsGRList**
  GRangesList output from getDifferentialAccessibleTiles
- **outDir**
  Desired output directory where bigBed files will be saved
- **verbose**
  Set TRUE to display additional messages. Default is FALSE.

**Value**

outList A List of output filepaths
Examples

```r
## Not run:
MOCHA::exportDifferentials(
  SampleTileObject = SampleTileMatrices,
  DifferentialsGRList,
  outDir = tempdir(),
  verbose = TRUE
)
## End(Not run)
```

Description

`exportMotifs` exports a motif set GRanges from running `addMotifSet(returnSTM=FALSE)` to bigBed file files for visualization in genome browsers.

Usage

```r
exportMotifs(
  SampleTileObject,
  motifsGRanges,
  motifSetName = "motifs",
  filterByOpenTiles = FALSE,
  outDir,
  verbose = FALSE
)
```

Arguments

- **SampleTileObject**
  The SummarizedExperiment object output from `getSampleTileMatrix`
- **motifsGRanges**
  A GRanges containing motif annotations, typically from `addMotifSet(returnSTM=FALSE)`
- **motifSetName**
  Optional, a name indicating the motif set. Used to name files in the specified outdir. Default is "motifs".
- **filterByOpenTiles**
  Boolean. If TRUE, a bigBed file will be exported for each cell population with motifs filtered to those occurring only in open tiles.
- **outDir**
  Desired output directory where bigBed files will be saved
- **verbose**
  Set TRUE to display additional messages. Default is FALSE.

Value

- **outList** A List of output filepaths
### Not run:
```
MOCHA::exportOpenTiles(
  SampleTileObject = SampleTileMatrices,
  motifsGRanges,
  motifSetName = "CISBP",
  filterByOpenTiles = FALSE,
  outDir = tempdir(),
  verbose = TRUE
)
```

### End(Not run)

---

### Description

`exportOpenTiles` exports the open tiles of a given cell population to bigBed file for visualization in genome browsers.

### Usage

```
exportOpenTiles(SampleTileObject, cellPopulation, outDir, verbose = FALSE)
```

### Arguments

- **SampleTileObject**
  The SummarizedExperiment object output from `getSampleTileMatrix`
- **cellPopulation**
  The name of the cell population to export
- **outDir**
  Desired output directory where bigBed files will be saved
- **verbose**
  Set TRUE to display additional messages. Default is FALSE.

### Value

- **outList** A List of output filepaths

### Examples

```
## Not run:
MOCHA::exportOpenTiles(
  SampleTileObject = SampleTileObject,
  cellPopulation,
  outDir = tempdir(),
  verbose = TRUE
)
```

## End(Not run)
Description

`exportSmoothedInsertions` Takes a SampleTileMatrix with linked insertion files and applies a smoothing filter (a rolling sum then rolling median) to the insertions, finally exporting the smoothed insertion files to bigwig format.

Usage

```r
exportSmoothedInsertions(
  SampleTileObj,
  cellPopulation,
  outDir = NULL,
  sumWidth = 10,
  medianWidth = 11,
  force = FALSE,
  slow = FALSE,
  verbose = FALSE
)
```

Arguments

- **SampleTileObj**: A MultiAssayExperiment or RangedSummarizedExperiment from MOCHA
- **cellPopulation**: A string denoting the cell population of interest
- **outDir**: Directory to write output bigwig files. Default is NULL, where the directory in `SampleTileObj@metadata$Directory` will be used.
- **sumWidth**: Window size for rolling sum in basepairs. Default is 10.
- **medianWidth**: Window size for rolling median in basepairs. Must be odd. Default is 11.
- **force**: Set TRUE to overwrite existing files. Default is FALSE.
- **slow**: Set TRUE to bypass optimisations and compute smoothing filter directly on the whole genome. May run slower and consume more RAM. Default is FALSE.
- **verbose**: Set TRUE to display additional messages. Default is FALSE.

Value

- **outPaths**: List of paths of exported insertion files

Examples

```r
## Not run:
# Depends on and manipulates files on filesystem
outPath <- MOCHA::exportSmoothedInsertions(
```
### Description

`extractRegion` will extract the coverage files created by `callOpenTiles` and return a specific region’s coverage.

### Usage

```r
evaluateRegion(
  SampleTileObj,  
  type = TRUE,     
  region,         
  cellPopulations = "ALL",  
  groupColumn = NULL,  
  subGroups = NULL,  
  sampleSpecific = FALSE,  
  approxLimit = 1e+05,  
  binSize = 250,  
  sliding = NULL,  
  numCores = 1,  
  verbose = FALSE
)
```

### Arguments

- **SampleTileObj**: The SummarizedExperiment object output from `getSampleTileMatrix`.
- **type**: Boolean. Default is true, and exports Coverage. If set to FALSE, exports Insertions.
- **region**: a GRanges object or vector or strings containing the regions of interest. Strings must be in the format "chr:start-end", e.g. "chr:1300-2222".
- **cellPopulations**: vector of strings. Cell subsets for which to call peaks. This list of group names must be identical to names that appear in the SampleTileObj. Optional, if `cellPopulations='ALL'`, then peak calling is done on all cell populations. Default is 'ALL'.
- **groupColumn**: Optional, the column containing sample group labels for returning coverage within sample groups. Default is NULL, all samples will be used.
**filterCoAccessibleLinks**

subGroups  a list of subgroup(s) within the groupColumn from the metadata. Optional, default is NULL, all labels within groupColumn will be used.

sampleSpecific  If TRUE, get a sample-specific count dataframe out. Default is FALSE, average across samples and get a dataframe out.

approxLimit  Optional limit to region size, where if region is larger than approxLimit basepairs, binning will be used. Default is 100000.

binSize  Optional numeric, size of bins in basepairs when binning is used. Default is 250.

sliding  Optional numeric. Default is NULL. This number is the size of the sliding window for generating average intensities.

numCores  integer. Number of cores to parallelize peak-calling across multiple cell populations

verbose  Set TRUE to display additional messages. Default is FALSE.

**Value**

countSE a SummarizedExperiment containing coverage for the given input cell populations.

**Examples**

```r
## Not run:
countSE <- MOCHA::extractRegion(
  SampleTileObj = SampleTileMatrices,
  cellPopulations = "ALL",
  region = "chr1:18137866-38139912",
  numCores = 30,
  sampleSpecific = FALSE
)
## End(Not run)
```

---

**Description**

filterCoAccessibleLinks will filter the output from getCoAccessibleLinks by a threshold, retaining links with a absolute correlation greater than the threshold. This function also adds the chr, start, and end site of each link to the output table.

**Usage**

`filterCoAccessibleLinks(TileCorr, threshold = 0.5)`
### Arguments

- **TileCorr**: The correlation table output from `getCoAccessibleLinks`
- **threshold**: Keep

### Description

Trained MOCHA models - LOESS and linear regression

### Usage

```r
finalModelObject
```

### Format

A list of lists containing 2 items: "Loess" and "Linear" each with "Total" "Max" and "Intercept"

- **Loess**: LOESS model
- **Linear**: Linear model

### Examples

```r
## Not run:
# links is the output of MOCHA::getCoAccessibleLinks
MOCHA::filterCoAccessibleLinks(links, threshold = 0.5)

## End(Not run)
```
### getAltTSS

Annotate Peaks falling in Transcription Start Sites (TSS) and identify alternatively regulated TSSs for each gene.

### Description

getAltTSS Pulls out all peaks that fall in TSS, annotates them with the name of gene, and identifies genes that have evidence for alternatively regulated TSSs, including both type i (only some of the open TSSs for a gene are significantly more (or less) accessible), and type ii (multiple TSSs are significant different, with some being more accessible and others less). Alternatively, this function will return all open TSSs with differential measurements if the returnAllTSS flag is set to TRUE.

### Usage

```r
getAltTSS(
  completeDAPs,
  returnAllTSS = FALSE,
  nuancedTSS = TRUE,
  nuancedTSSGap = 150,
  threshold = 0.2,
  TxDb,
  OrgDb
)
```

### Arguments

- **completeDAPs**: GRanges object that contains the differential measurements across all peaks (unfiltered DAPs). Will also work with data.frame or data.table version of a GRanges object. If you want alternatively regulated TSSs, the object must include a column names 'FDR', and 'Log2FC_C', which is standard for MOCHA differentials.
- **returnAllTSS**: Flag to return all TSSs with DAPs measurements, without filtering for alternative TSS usage. If multiple TSSs fall within the same tile, then that tile will be repeated for each TSS.
- **nuancedTSS**: True/False flag to determine if alternative TSS genes should be filtered out if all their differential TSS usage falls within too small of a range. Default is TRUE.
- **nuancedTSSGap**: Minimum distance between TSSs needed for them to considered distinctly regulated TSSs. If two TSSs are too close, it is unclear and highly unlikely that ATAC data can distinguish between them. Default is 150 bp.
- **threshold**: FDR Threshold for determining significant vs non-significant changes in accessibility. Following MOCHA's standards, default is 0.2.
- **TxDb**: The TxDb-class transcript annotation package for your organism (e.g. "TxDb.Hsapiens.UCSC.hg38.refGene"). This must be installed. See Bioconductor AnnotationData Packages.
- **OrgDb**: The OrgDb-class genome wide annotation package for your organism (e.g. "org.Hs.eg.db"). This must be installed. See Bioconductor AnnotationData Packages.
getAnnotationDbFromInstalledPkgname

getAnnotationDbFromInstalledPkgname Loads and attaches an installed TxDb or OrgDb-class Annotation database package.

Description

See getBSgenome

Usage

getAnnotationDbFromInstalledPkgname(dbName, type)

Arguments

dbName

Exact name of installed annotation data package.

type

Expected class of the annotation data package, must be either "OrgDb" or "TxDb".

Value

the loaded Annotation database object.

getCellPopMatrix

getCellPopMatrix pulls out the SampleTileMatrix of tiles called in one given cell population.

Usage

ggetCellPopMatrix(
    SampleTileObj,
    cellPopulation,
    dropSamples = TRUE,
    NAtoZero = TRUE
)

Value

tpeaks A GRanges containing annotated peaks falling in TSS
**getCellTypes**

**Arguments**

- **SampleTileObj**: The output from `getSampleTileMatrix`, a `SummarizedExperiment` of pseudobulk intensities across all tiles & cell types.
- **cellPopulation**: The cell population you want to pull out.
- **dropSamples**: Boolean flag to determine whether to drop samples that were too small for peak calling.
- **NAtoZero**: Boolean flag to determine whether to replace NAs with zero

**Value**

- `sampleTileMatrix`: a matrix of samples by called tiles for a given cell population.

---

**getCellTypeTiles**

**Description**

getCellTypeTiles returns a GRanges object of all tiles called for a certain cell type.

**Usage**

ggetCellTypeTiles(object, cellType)
**getCoAccessibleLinks**

**Arguments**

- **object**
  A SampleTileObject.

- **cellType**
  A string describing one cell type.

**Value**

a vector of cell type names.

---

**getCoAccessibleLinks**

**Description**

getCoAccessibleLinks takes an input set of regions (tiles) and finds co-accessible neighboring regions within a window. Co-accessibility is defined as the correlation between two region intensity (openness) across samples.

**Usage**

```r
getCoAccessibleLinks(
  SampleTileObj,
  cellPopulation = "All",
  regions,
  chrChunks = 1,
  windowSize = 1 * 10^6,
  numCores = 1,
  ZI = TRUE,
  approximateTile = FALSE,
  verbose = FALSE
)
```

**Arguments**

- **SampleTileObj**
  The SummarizedExperiment object output from getSampleTileMatrix containing your sample-tile matrices

- **cellPopulation**
  A string denoting the cell population of interest, which must be present in SampleTileObj

- **regions**
  a GRanges object or vector or strings containing the regions on which to compute co-accessible links. Strings must be in the format "chr:start-end", e.g. "chr4:1300-2222". Can be the output from getDifferentialAccessibleTiles.

- **chrChunks**
  This functions subsets by groups of chromosome, and then parallelizes within each group of chromosomes when running correlations. This method keeps memory low. To speed things up on high performing platforms, you can chunk out more than one chromosome at a time. Default is chrChunks = 1, so only one chromosome at a time.
**getCoverage**

 windowSize  
 the size of the window, in basepairs, around each input region to search for co-accessible links

 numCores  
 Optional, the number of cores to use with multiprocessing. Default is 1.

 ZI  
 boolean flag that enables zero-inflated (ZI) Spearman correlations to be used. Default is TRUE. If FALSE, skip zero-inflation and calculate the normal Spearman.

 approximateTile  
 If set to TRUE, it will use all tiles that overlap with the regions given, instead of finding an exact match to the regions variable. Default is FALSE.

 verbose  
 Set TRUE to display additional messages. Default is FALSE.

**Details**

The technical details of the zero-inflated correlation can be found here:


while the implementation (scHOT R package), can be found here: http://www.bioconductor.org/packages/release/bioc/html/scHOT.html

**Value**

TileCorr A data.table correlation matrix

---

**Usage**

getCoverage(
  popFrags,
  normFactor,
  TxBdb,
  cl,
  filterEmpty = FALSE,
  verbose = FALSE
)
getDifferentialAccessibleTiles

Arguments

popFrags GRangesList of fragments for all sample/cell populations
normFactor Normalization factor. Can be either be one, in which case all coverage files will be normalized by the same value, or the same length as the GRangesList
TxDb The TxDb-class transcript annotation package for your organism (e.g. "TxDb.Hsapiens.UCSC.hg38.refGene". This must be installed. See Bioconductor AnnotationData Packages.
cl cl argument to pblapply
filterEmpty True/False flag on whether or not to carry forward regions without coverage.
verbose Boolean variable to determine verbosity of output.

Value

popCounts A GRangesList of coverage for each sample and cell population

Description

getDifferentialAccessibleTiles allows you to determine whether regions of chromatin are differentially accessible between groups by conducting a test

Usage

ggetDifferentialAccessibleTiles(
  SampleTileObj, 
cellPopulation, 
groupColumn, 
foreground, 
background, 
signalThreshold = 12, 
minZeroDiff = 0.5, 
fdrToDisplay = 0.2, 
outputGRanges = TRUE, 
numCores = 1, 
verbose = FALSE
)

Arguments

SampleTileObj The SummarizedExperiment object output from getSampleTileMatrix
cellPopulation A string denoting the cell population of interest
groupColumn The column containing sample group labels
**getDifferentialAccessibleTiles**

foreground: The foreground group of samples for differential comparison

background: The background group of samples for differential comparison

signalThreshold: Minimum median intensity required to keep tiles for differential testing to increase statistical power in small sample cohorts. Default is 12.

minZeroDiff: Minimum difference in average dropout rates across groups require to keep tiles for differential testing. Default is 0.5 (50%).

fdrToDisplay: False-discovery rate used only for standard output messaging. Default is 0.2.

outputGRanges: Outputs a GRanges if TRUE and a data.frame if FALSE. Default is TRUE.

numCores: The number of cores to use with multiprocessing. Default is 1.

verbose: Set TRUE to display additional messages. Default is FALSE.

**Value**

full_results: The differential accessibility results as a GRanges or matrix data.frame depending on the flag `outputGRanges`.

**Examples**

```r
## Not run:
cellPopulation <- "MAIT"
foreground <- "Positive"
background <- "Negative"
# Standard output will display the number of tiles found below a false-discovery rate threshold.
# This parameter does not filter results and only affects the aforementioned message.
fdrToDisplay <- 0.2
# Choose to output a GRanges or data.frame.
# Default is TRUE
outputGRanges <- TRUE
# SampleTileMatrices is the output of MOCHA::getSampleTileMatrix
differentials <- MOCHA::getDifferentialAccessibleTiles(
  SampleTileObj = SampleTileMatrices,
  cellPopulation = cellPopulation,
  groupColumn = groupColumn,
  foreground = foreground,
  background = background,
  fdrToDisplay = fdrToDisplay,
  outputGRanges = outputGRanges,
  numCores = numCores
)

## End(Not run)
```
Description

getIntensityThreshold takes the output of peak calling with callOpenTiles and creates sample-tile matrices containing the signal intensity at each tile.

Usage

getIntensityThreshold(
  TSAM,
  cellPopulations = "all",
  type = "mean",
  returnPlots = TRUE,
  verbose = FALSE
)

Arguments

- **TSAM**: a SummarizedExperiment object generated by MOCHA.
- **cellPopulations**: vector of strings. Cell subsets found in the TSAM, or the word 'All' if all should be used.
- **type**: string. Describes the type of metric to be used. Options include median or mean.
- **returnPlots**: Boolean. Default is TRUE and returns a plot of
- **verbose**: Set TRUE to display additional messages. Default is FALSE.

Value

- plot object

getModelValues

Description

'r lifecycle::badge("deprecated")': This function is deprecated - improved modeling functions can be found in the package "ChAI" at https://github.com/aifimmunology/ChAI getModelValues Pull out a data.frame of model values (slope, significance, and std.error) for a given factor from the SummarizedExperiment output of runZIGLMM.

Usage

getModelValues(object, specificVariable)
**getPopFrags**

**Arguments**

- **object**: A SummarizedExperiment object generated from runZIGLMM.
- **specificVariable**: A string, describing the factor of influence.

**Value**

A data.frame of slopes, significance, and standard error for one factor.

**Examples**

```r
## Not run:
age_df <- getModelValues(runZIGLMM_output, "Age")
## End(Not run)
```

**Description**

getPopFrags returns a list of sample-specific fragments per cell population as a GRangesList.

**Usage**

```r
getPopFrags(
  ArchRProj, 
  cellPopLabel, 
  cellSubsets = "ALL", 
  poolSamples = FALSE, 
  numCores = 1, 
  verbose = FALSE
)
```

**Arguments**

- **ArchRProj**: The ArchR Project.
- **cellPopLabel**: The name of the metadata column of the ArchR Project that contains the populations of cells you want to extract fragments from.
- **cellSubsets**: Default is 'ALL'. If you want to export only some populations, then give it a list of group names. This needs to be unique - no duplicated names. This list of group names must be identical to names that appear in the given cellPopLabel metadata column of the ArchR Project.
- **poolSamples**: Set TRUE to pool sample-specific fragments by cell population. By default this is FALSE and sample-specific fragments are returned.
- **numCores**: Number of cores to use.
- **verbose**: Set TRUE to display additional messages. Default is FALSE.
**Value**

A list of GRanges containing fragments. Each GRanges corresponds to a population defined by cellSubsets and sample.

**getPromoterGenes**

**Description**

gPromoterGenes Takes a rowRanges from annotateTiles and extracts a unique list of genes.

**Usage**

gPromoterGenes(GRangesObj)

**Arguments**

GRangesObj a GRanges object with a metadata column for tileType and Gene.

**Value**

vector of strings with gene names.

**getSampleCellTypeMetadata**

**Description**

gSampleCellTypeMetadata Extract Sample-celltype specific metadata like fragment number, cell counts, and

**Usage**

gSampleCellTypeMetadata(object)

**Arguments**

object tileResults object from callOpenTiles or SummarizedExperiment from getSampleTileMatrix

**Value**

a SummarizedExperiment where each assay is a different type of metadata.
getSampleTileMatrix

Description

getSampleTileMatrix takes the output of peak calling with callOpenTiles and creates sample-tile matrices containing the signal intensity at each tile.

Usage

getSampleTileMatrix(
  tileResults,
  cellPopulations = "ALL",
  groupColumn = NULL,
  threshold = 0.2,
  numCores = 1,
  verbose = FALSE
)

Arguments

tileResults a MultiAssayExperiment returned by callOpenTiles containing containing peak calling results.

cellPopulations vector of strings. Cell subsets in TileResults for which to generate sample-tile matrices. This list of group names must be identical to names that appear in the ArchRProject metadata. If cellPopulations='ALL', then peak calling is done on all cell populations in the ArchR project metadata. Default is 'ALL'.

groupColumn Optional, the column containing sample group labels for determining consensus tiles within sample groups. Default is NULL, all samples will be used for determining consensus tiles.

threshold Threshold for consensus tiles, the minimum % of samples (within a sample group, if groupColumn is set) that a peak must be called in to be retained. If set to 0, retain the union of all samples’ peaks (this is equivalent to a threshold of 1/numSamples). It is recommended to tune this parameter to omit potentially spurious peaks.

numCores Optional, the number of cores to use with multiprocessing. Default is 1.

verbose Set TRUE to display additional messages. Default is FALSE.

Value

SampleTileMatrices a MultiAssayExperiment containing a sample-tile intensity matrix for each cell population
getSequencingBias

Examples

# Starting from GRangesList
if (require(BSgenome.Hsapiens.UCSC.hg19) && require(TxDb.Hsapiens.UCSC.hg38.refGene) && require(org.Hs.eg.db)) {
  tiles <- MOCHA::callOpenTiles(
    ATACFragments = MOCHA::exampleFragments,
    cellColData = MOCHA::exampleCellColData,
    blacklist = MOCHA::exampleBlackList,
    genome = "BSgenome.Hsapiens.UCSC.hg19",
    TxDb = "TxDb.Hsapiens.UCSC.hg38.refGene",
    Org = "org.Hs.eg.db",
    outDir = tempdir(),
    cellPopLabel = "Clusters",
    cellPopulations = c("C2", "C5"),
    numCores = 1
  )
  SampleTileMatrices <- MOCHA::getSampleTileMatrix(
    tiles,
    cellPopulations = c("C2", "C5"),
    threshold = 0 # Take union of all samples' open tiles
  )
}

getSequencingBias  getSequencingBias

Description

generateBias takes the output of peak calling with callOpenTiles and creates sample-tile matrices containing the signal intensity at each tile.

Usage

generateBias(
  SampleTileObj,
  cellPopulations = "all",
  cellPopulation,
  groupColumn,
  foreground,
  background,
  verbose = TRUE
)
GRangesToString

Arguments

SampleTileObj a SummarizedExperiment object generated by MOCHA

cellPopulations vector of strings. Cell subsets found in the TSAM, or the word 'All' if all should be used.

cellPopulation A string denoting the cell population of interest

groupColumn The column containing sample group labels

foreground The foreground group of samples for differential comparison

background The background group of samples for differential comparison

verbose Set TRUE to display additional messages. Default is FALSE.

Value

plot object

GRangesToString

Converts a GRanges object to a string in the format 'chr1:100-200'

Description

GRangesToString Turns a GRanges Object into a list of strings in the format chr1:100-200

Usage

GRangesToString(GR_obj)

Arguments

GR_obj the GRanges object to convert to a string

Value

A string or list of strings in the format 'chr1:100-200' representing ranges in the input GRanges
mergeTileResults

Description

mergeTileResults merges a list of tileResults that each contain unique samples into a single object encompassing all samples. Only cell populations shared among all input tileResults will be retained. This function can merge MultiAssayExperiment objects from callOpenTiles that are created with the same TxDb, OrgDb, and Genome assembly.

Usage

mergeTileResults(tileResultsList, numCores = 1, verbose = TRUE)

Arguments

- tileResultsList: List of MultiAssayExperiments objects returned by callOpenTiles containing containing peak calling results.
- numCores: Optional, the number of cores to use with multiprocessing. Default is 1.
- verbose: Set TRUE to display additional messages. Default is FALSE.

Value

tileResults a single MultiAssayExperiment containing a sample-tile intensity matrix for each sample and common cell population in the input tileResultsList.

Examples

```r
## Not run:
# Depends on local MOCHA tileResults
MOCHA::mergeTileResults(
  list(tileResultsCelltypesABC, tileResultsCelltypesBCD)
)
## End(Not run)
```
MotifEnrichment

Description
Test for enrichment of motifs within Group1 against a background Group2 using a hypergeometric t-test.

Usage
MotifEnrichment(Group1, Group2, motifPosList, type = NULL)

Arguments
- Group1: A GRanges object, such as a set of significant differential tiles.
- Group2: A GRanges object containing background regions, non-overlapping with Group1
- motifPosList: A GRangesList of motifs and positions for each motif. Must be named for each motif.
- type: Optional, name of a metadata column in Group1 and Group2 to test for enrichment the number of unique entries in column given by 'type'. Default is NULL, which tests the number of Ranges.

Value
A data.frame containing enrichment for each group

MotifSetEnrichmentAnalysis

Description
This analogous to Gene Set Enrichment Analysis. Instead of testing for enrichment of a geneset with a given gene set in a pathway, we are testing the enrichment of a given TF motif set against a motif set downstream of a multiple ligands. If there is enrichment, it's a sign that that ligand could drive that set of motifs.

Usage
MotifSetEnrichmentAnalysis(
ligandTFMatrix, motifEnrichmentDF, motifColumn, ligands, statColumn,
statThreshold,
annotationName = "CellType",
annotation = "none",
umCores = 1,
verbose = FALSE
)

Arguments

ligandTFMatrix  NicheNet Ligand-TF matrix
motifEnrichmentDF
  Dataframe (unfiltered) from ArchR’s peakAnnoEnrich step. Expected to have a
column with motif names, and a column with the -log10 adjusted p-values.
motifColumn
  Column name within the motifEnrichmentDF that has motif names.
ligands
  Vector of ligands to test
statColumn
  Column name in motifEnrichmentDF containing the statistic to test
statThreshold
  Significance threshold used to select significant motif set
annotationName
  Optional column name for the annotation. Default is "CellType".
annotation
  Optional annotation value added to all rows of the output motif dataframe. Can
  be character vector or numeric. Default is "none".
numCores
  The number of cores to use with multiprocessing. Default is 1.
verbose
  Set TRUE to display additional messages. Default is FALSE.

Value

specDF A dataframe containing enrichment analysis results

Description

packMOCHA combines a MOCHA object (Sample-Tile Matrix or tileResults) with its saved coverage
tracks into a single zip archive. This allows MOCHA objects and the necessary coverage files for
plotting to be shared to other file systems. See also: unpackMOCHA

Usage

packMOCHA(MOCHAObj, zipfile, verbose = FALSE)

Arguments

MOCHAObj
  A MultiAssayExperiment or RangedSummarizedExperiment, from MOCHA
zipfile
  Filename and path of the zip archive.
verbose
  Set TRUE to display additional messages. Default is FALSE.
pilotLMEM

Value

zipfile Path to zip archive.

Examples

```r
## Not run:
# Depends on and manipulates files on filesystem
myOutputDir <- "/home/documents/MOCHA_out"
zipPath <- MOCHA::packMOCHA(
  tileResults, zipfile = file.path(myOutputDir, "testzip.zip")
)
## End(Not run)
```

---

**pilotLMEM**

| Execute a pilot run of single linear model on a subset of data |

Description

`r lifecycle::badge("deprecated")` This function is deprecated - improved modeling functions can be found in the package "ChAI" at https://github.com/aifimmunology/ChAI  

pilotLMEM  

Runs linear mixed-effects modeling for continuous, non-zero inflated data using `lmer`

Usage

```r
pilotLMEM(
  ExperimentObj, 
  assayName, 
  modelFormula, 
  pilotIndices = 1:10, 
  verbose = FALSE
)
```

Arguments

- **ExperimentObj** A SummarizedExperiment-type object generated from chromVAR, makePseudobulkRNA, or other. Objects from getSampleTileMatrix can work, but we recommend runZIGLMM for those objects, **not** runLMEM>
- **assayName** a character string, matching the name of an assay within the SummarizedExperiment. The assay named will be used for modeling.
- **modelFormula** The formula to use with `lmerTest::lmer`, in the format (exp ~ factors). All factors must be found in column names of the ExperimentObj metadata.
- **pilotIndices** A vector of integers defining the subset of the ExperimentObj matrix. Default is 1:10.
- **verbose** Set TRUE to display additional messages. Default is FALSE.
pilotZIGLMM

Value

modelList a list of outputs from lmerTest::lmer

pilotZIGLMM  Execute a pilot run of model on a subset of data

Description

`r lifecycle::badge("deprecated")` This function is deprecated - improved modeling functions can be found in the package "ChAI" at https://github.com/aifimmunology/ChAI

pilotLMEM Runs linear mixed-effects modeling for zero inflated data using glmmTMB. TryCatch will catch errors, and return the error and dataframe for troubleshooting.

Usage

```r
pilotZIGLMM(
  TSAM_Object,
  cellPopulation = NULL,
  continuousFormula = NULL,
  ziformula = NULL,
  zi_threshold = 0,
  verbose = FALSE,
  pilotIndices = 1:10
)
```

Arguments

- **TSAM_Object**: A SummarizedExperiment object generated from getSampleTileMatrix, chromVAR, or other.
- **cellPopulation**: A single cell population on which to run this pilot model
- **continuousFormula**: The formula, see glmmTMB. Combined fixed and random effects formula, following lme4 syntax.
- **ziformula**: The zero-inflated formula, see glmmTMB. a one-sided (i.e., no response variable) formula for zero-inflation combining fixed and random effects: the default ~0 specifies no zero-inflation. Specifying ~. sets the zero-inflation formula identical to the right-hand side of formula (i.e., the conditional effects formula); terms can also be added or subtracted. When using ~. as the zero-inflation formula in models where the conditional effects formula contains an offset term, the offset term will automatically be dropped. The zero-inflation model uses a logit link.
- **zi_threshold**: Zero-inflated threshold (range = 0-1), representing the fraction of samples with zeros. When the percentage of zeros in the tile is between 0 and zi_threshold, samples with zeroes are dropped and only the continuous formula is used. Use this parameter at your own risk. Default is 0.
- **verbose**: Set TRUE to display additional messages. Default is FALSE.
- **pilotIndices**: A vector of integers defining the subset of the ExperimentObj matrix. Default is 1:10.
plotConsensus

Value
modelList a list of outputs from glmmTMB::glmmTMB

Description
plotConsensus Extracts the peak reproducibility and generates a heuristic plots that can be used to determine the reproducibility threshold used within getSampleTileMatrix.

Usage
plotConsensus(
  tileObject, cellPopulations = "All",
  groupColumn = NULL, returnPlotList = FALSE,
  returnDFs = FALSE, numCores = 1
)

Arguments
tileObject A MultiAssayExperiment object from callOpenTiles,
cellPopulations the cell populations you want to visualize.
groupColumn Optional parameter, same as in getSampleTileMatrix, which defines whether you want to plot reproducibility within each
returnPlotList Instead of one plot with all celltypes/conditions, it returns a list of plots for each
cell types
returnDFs Instead of a plot, returns a data.frame of the reproducibility across samples. If
set to false, then it plots the data.frame instead of returning it.
numCores Number of cores to multithread over.

Value
SampleTileObj the input data structure with added gene annotations.
Description

plotIntensityDistribution Plots the distribution of sample-tile intensities for a given cell type.

Usage

plotIntensityDistribution(
  TSAM_object,
  cellPopulation,
  returnDF = FALSE,
  density = TRUE
)

plotIntensityDistribution(
  TSAM_object,
  cellPopulation,
  returnDF = FALSE,
  density = TRUE
)

Arguments

TSAM_object SummarizedExperiment from getSampleTileMatrix

cellPopulation Cell type names (assay name) within the TSAM_object

returnDF If TRUE, return the data frame without plotting. Default is FALSE.

density Boolean to determine whether to plot density or histogram. Default is TRUE (plots density).

Value

data.frame or ggplot histogram.

data.frame or ggplot histogram.
plotRegion

Description

plotRegion Plots the region that you’ve summarized across all cell groupings (groups=initial getPopFrgs() split) with optional motif overlay, chromosome position ideogram, and additional GRanges tracks. If plotting motif overlay, ensure that motif annotations have been added to your counts SummarizedExperiment. A basic plot can be rendered with just a counts SummarizedExperiment, but additional formatting arguments allow for further customization. Note that to show specific genes with the option ‘whichGenes’ the RMariaDB package must be installed.

Usage

plotRegion(
  countSE,
  plotType = "area",
  base_size = 12,
  counts_color = NULL,
  range_label_size = 2,
  legend.position = NULL,
  legendRatio = 0.25,
  facet_label_side = "top",
  counts_color_var = "Groups",
  counts_group_colors = NULL,
  counts_theme_ls = NULL,
  motifSetName = NULL,
  motif_y_space_factor = 4,
  motif_stagger_labels_y = FALSE,
  motif_weights = NULL,
  motif_weight_name = "Motif Weight",
  motif_weight_colors = c(darkblue = -10, gray = 0, darkred = 10),
  motif_lab_size = 1,
  motif_lab_alpha = 0.25,
  motif_line_alpha = 0.25,
  motif_line_size = 0.75,
  showGene = TRUE,
  whichGenes = NULL,
  monotoneGenes = FALSE,
  db_id_col = "REFSEQ",
  collapseGenes = FALSE,
  gene_theme_ls = NULL,
  additionalGRangesTrack = NULL,
  linkdf = NULL,
  showIdeogram = TRUE,
  ideogram_genome = "hg19",
  relativeHeights = c(Chr = 0.9, ‘Normalized Counts’ = 7, Links = 1.5, Genes = 2,
plotRegion

AdditionalGRanges = 4.5),
verbose = FALSE
)

Arguments

countSE A SummarizedExperiment from MOCHA::getCoverage

plotType Options include 'overlaid', 'area', 'line', or 'RidgePlot'. default is 'area', which will plot a separate track for each group with the area filled in under the curve. Setting plotType to 'overlaid' will overlay count plot histograms across samples, instead of faceting out separately. Setting plotType to 'RidgePlot' will generate a RidgePlot across all groups.

base_size Numeric, default 12. Global plot base text size parameter

counts_color Optional color palette. A named vector of color values where names are unique values in the 'color_var' column

range_label_size Numeric value, default 4. Text size for the y-axis range label

legend.position Any acceptable 'legend.position' argument to theme(). Default NULL will place legend for overlaid plots at (0.8,0.8), or to the "right" for faceted plots.

legendRatio Ratio of width or height of the main plot to the legend. Useful if the legend is to large. If only used when legend.position is set to top, bottom, left, or right.

facet_label_side Direction character value, default "top". Can also be "right", "left", or "bottom". Position of facet label.

counts_color_var Character value, default "Groups". Column name from countdf to use to color counts plots. Only used if counts_group_colors provided

counts_group_colors Optional named color vector. Values as colors, names are levels of 'counts_color_var'. If provided, will color the plots specifically using 'scale_color_manual()'

counts_theme_ls A list of named theme arguments passed to theme(). For example, 'list(axis.ticks = element_blank())'. Default NULL will use '.counts_plot_default_theme'.

motifSetName The name of the motif set in ArchRProj to use for annotation. Example: 'JasparMotifs'

motif_space_factor A factor for vertical spacing between motif labels. Default 4. Increase to make labels farther apart, decrease to make labels closer.

motif_stagger_labels_y = FALSE Logical value, default FALSE. If TRUE, will stagger motif labels in adjacent columns in the vertical direction

motif_weights Optional numeric vector, default NULL. If provided will be used to color motif labels by the weighted values
motif_weight_name
Character value, default "Motif Weight". Used to label the legend for motif colors.

motif_weight_colors
Named numeric vector. Names should be color values and breaks should be the corresponding values of motif_weights. Values outside the highest and lowest value will appear as max or min defined color value.

motif_lab_size
Numeric value, default 1. Size of motif labels.

motif_lab_alpha
Numeric value, default 0.25. Alpha for motif labels.

motif_line_alpha
Numeric value, default 0.25. Alpha for motif lines.

motif_line_size
Numeric value, default 1. Size of motif lines.

showGene
Logical value, default TRUE. Whether or not the gene track should be plotted.

whichGenes
Name of gene for plotting this specific gene in region.

monotoneGenes
Boolean. Determines whether to color-code genes by gene name, or to set them all to dark gray.

db_id_col
Character value. Column in 'OrgDb' containing the output id for 'whichGenes' plotting. Default "REFSEQ".

collapseGenes
Options include 'collapseAll', 'longestTx', or 'None'. Default 'None' will plot the expanded view of the reference genes, 'collapseAll' if you want collapse the gene tracks into one, and 'longestTx' will only plot the longest transcript of each gene.

gene_theme_ls
Named list of parameters passed to 'theme()' for the gene plot. Default NULL will use '.gene_plot_theme'.

additionalGRangesTrack
A GRanges object containing additional track plot data.

linkdf
A dataframe with co-accessible links to display as an additional track.

showIdeogram
Logical value, default TRUE. If TRUE plots the chromosome ideogram at the top of the multi-track plot.

ideogram_genome
Character value, a genome name for the ideogram plot. Default 'hg19'.

relativeHeights
Named numeric vector of relative heights for each of the 4 track plots to enable clean visualization when there are many tracks. Unused tracks will be ignored. Default value = c('Chr' = 0.9, 'Normalized Counts' = 7, 'Genes'= 2, 'AdditionalGRanges' = 4.5)

verbose
Set TRUE to display additional messages. Default is FALSE.

Value
The input ggplot object with motif labels overlaid.
Examples

### Not run:

```r
# my_count_SE is a counts data frame generated by extractRegion()

# Simple counts + ideogram + all genes:
plotRegion(countSE = my_count_SE)

# Motif overlay for a project my_proj containing "JasparMotifs" annotations:
plotRegion(
  countSE = my_count_SE, motifSetName = "JasparMotifs",
  motif_lab_alpha = 1, motif_line_alpha = 1
)

# Motif overlay w/ weights:
plotRegion(
  countSE = my_count_SE, motifSetName = "JasparMotifs", motif_lab_alpha = 1,
  motif_line_alpha = 1, motif_weights = my_enrichment_weights
)

### End(Not run)
```

renameCellTypes renameCellTypes

Description

renameCellTypes Allows you to modify the cell type names for a MOCHA SampleTileObject, from the assay names, GRanges column names, and summarizedData (within the metadata), all at once.

Usage

```r
renameCellTypes(MOCHAObject, oldNames, newNames)
```

Arguments

- **MOCHAObject** A RangedSummarizedExperiment,
- **oldNames** A list of cell type names that you want to change.
- **newNames** A list of new cell type names to replace the old names with.

Value

A MOCHA SampleTile object with new cell types.
runLMEM

Run Linear Mixed-Effects Modeling for continuous, non-zero inflated data

Description

runLMEM Runs linear mixed-effects modeling for continuous, non-zero inflated data using \texttt{lmer}

Usage

runLMEM(
  ExperimentObj,
  assayName,
  modelFormula,
  initialSampling = 5,
  verbose = FALSE,
  numCores = 2
)

Arguments

- **ExperimentObj** A SummarizedExperiment object generated from getSampleTileMatrix, chromVAR, or other. It is expected to contain only one assay, or only the first assay will be used for the model. Data should not be zero-inflated.
- **assayName** The name of the assay to model within the SummarizedExperiment.
- **modelFormula** The formula to use with \texttt{lmerTest::lmer}, in the format (exp $\sim$ factors). All factors must be found in column names of the ExperimentObj metadata. modelFormula must start with 'exp' as the response. See \texttt{lmer}.
- **initialSampling** Size of data to use for pilot
- **verbose** Set TRUE to display additional messages. Default is FALSE.
- **numCores** integer. Number of cores to parallelize across.

Value

results a SummarizedExperiment containing LMEM results. Assays are metrics related to the model coefficients, including the Estimate, Std_Error, df, t_value, p_value. Within each assay, each row corresponds to each row of the SummarizedExperiment and columns correspond to each fixed effect variable within the model. Any row metadata from the ExperimentObject (see rowData(ExperimentObj)) is preserved in the output. The Residual matrix and the variance of the random effects are saved in the metadata slot of the output.
Examples

```r
## Not run:
modelList <- runLMEM(ExperimentObj,
  assayName = names(ExperimentObj)[[1]],
  modelFormula = NULL,
  initialSampling = 5,
  verbose = FALSE,
  numCores = 1
)
## End(Not run)
```

runZIGLMM

*Run Zero-inflated Generalized Linear Mixed Modeling on pseudobulked scATAC data*

Description

 `'r lifecycle::badge("deprecated")`: This function is deprecated - improved modeling functions can be found in the package "ChAI" at https://github.com/aifimmunology/ChAI. `runZIGLMM` runs linear mixed-effects modeling for zero-inflated data using `glmmTMB`.

Usage

```r
runZIGLMM(
  TSAM_Object,
  cellPopulation = "all",
  continuousFormula = NULL,
  ziformula = NULL,
  zi_threshold = 0,
  initialSampling = 5,
  verbose = FALSE,
  numCores = 1
)
```

Arguments

- `TSAM_Object` A SummarizedExperiment object generated from `getSampleTileMatrix`.
- `cellPopulation` Name of a cell type(s), or 'all'. The function will combine the cell types mentioned into one matrix before running the model.
- `continuousFormula` The formula for the continuous data that should be used within `glmmTMB`. It should be in the format `(exp ~ factors)`. All factors must be found in column names of the TSAM_Object metadata, except for `CellType`, `FragNumber` and `CellCount`, which will be extracted from the TSAM_Object. `modelFormula` must start with 'exp' as the response. See `glmmTMB`.
StringsToGRanges

**ziformula**  The formula for the zero-inflated data that should be used within glmmTMB. It should be in the format (~ factors). All factors must be found in column names of the TSAM_Object colData metadata, except for CellType, FragNumber and CellCount, which will be extracted from the TSAM_Object.

**zi_threshold**  Zero-inflated threshold (range = 0-1), representing the fraction of samples with zeros. When the percentage of zeros in the tile is between 0 and zi_threshold, samples with zeroes are dropped and only the continuous formula is used. Use this parameter at your own risk. Default is 0.

**initialSampling**  Size of data to use for pilot

**verbose**  Set TRUE to display additional messages. Default is FALSE.

**numCores**  integer. Number of cores to parallelize across.

**Value**

results a SummarizedExperiment containing LMEM results

**Examples**

```r
## Not run:
modellist <- runZIGLMM(STM[,c(1:1000), ],
cellPopulation = "CD16 Mono",
continuousFormula = exp ~ Age + Sex + days_since_symptoms + (1 | PTID),
ziFormula = ~ FragNumber + Age,
verbose = TRUE,
numCores = 35
)
## End(Not run)
```

---

**StringsToGRanges**

**Description**

StringsToGRanges Turns a list of strings in the format chr1:100-200 into a GRanges object

**Usage**

StringsToGRanges(regionString)

**Arguments**

regionString  A string or list of strings each in the format chr1:100-200

**Value**

a GRanges object with ranges representing the input string(s)
subsetMOCHAObject

Description

subsetMOCHAObject subsets a tileResults-type object (from callOpenTiles), or a SummarizedExperiment-type object (from getSampleTileMatrix), either by cell type or sample metadata.

Usage

subsetMOCHAObject(
  Object,
  subsetBy,
  groupList,
  removeNA = TRUE,
  subsetPeaks = TRUE,
  verbose = FALSE
)

Arguments

Object A MultiAssayExperiment or RangedSummarizedExperiment,
subsetBy The variable to subset by. Can either be ‘celltype’, or a column from the sample metadata (see 'colData(Object)').
groupList the list of cell type names or sample-associated data that should be used to subset the Object
removeNA If TRUE, removes groups in groupList that are NA. If FALSE, keep groups that are NA.
subsetPeaks If ‘subsetBy’ = ‘celltype’, subset the tile set to tiles only called in those cell types. Default is TRUE.
verbose Set TRUE to display additional messages. Default is FALSE.

Value

Object the input Object, filtered down to either the cell type or samples desired.
Description

testCoAccessibility takes an input set of tile pairs and tests whether they are significantly different compared to random, non-overlapping background set.

Usage

testCoAccessibility(
  SampleTileObj,
  tile1,
  tile2,
  numCores = 1,
  ZI = TRUE,
  backNumber = 1000,
  calcPValue = TRUE,
  returnBackGround = FALSE,
  verbose = TRUE
)

Arguments

SampleTileObj
  The SummarizedExperiment object output from getSampleTileMatrix containing your sample-tile matrices

tile1
  vector of indices or tile names (chrX:100-2000) for tile pairs to test (first tile in each pair)

tile2
  vector of indices or tile names (chrX:100-2000) for tile pairs to test (second tile in each pair)

numCores
  Optional, the number of cores to use with multiprocessing. Default is 1.

ZI
  boolean flag that enables zero-inflated (ZI) Spearman correlations to be used. Default is TRUE. If FALSE, skip zero-inflation and calculate the normal Spearman.

backNumber
  number of background pairs. Default is 1000.

calcPValue
  Boolean, if TRUE calculate p-values. Default is TRUE.

returnBackGround
  Boolean, if TRUE return the background correlations as well as foreground. Default is FALSE.

verbose
  Set TRUE to display additional messages. Default is FALSE.

Value

foreGround A data.frame with Tile1, Tile2, Correlation, and p-value for that correlation compared to the background
unpackMOCHA will unpack a zip archive created by `unpackMOCHA`, setting the stored MOCHA object’s stored directory path to the new location. See also: `packMOCHA`

### Usage

```r
unpackMOCHA(zipfile, exdir, verbose = FALSE)
```

### Arguments

- `zipfile`: Filepath to the packed MOCHA object.
- `exdir`: The path to the external directory where you want to unpack the MOCHA object.
- `verbose`: Display additional messages. Default is `FALSE`.

### Value

- `MOCHAObj` the MOCHA object (tileResults or Sample-Tile Matrix)

### Examples

```r
## Not run:
# Depends on files existing on your system
MOCHA::unpackMOCHA(zipfile = "/mochaobj.zip", exdir = "/newMOCHAdir")
## End(Not run)
```

---

varZIGLMM

Zero-inflated Variance Decomposition for pseudobulked scATAC data

### Description

This function is deprecated - improved modeling functions can be found in the package “ChAI” at https://github.com/aifimmunology/ChAI varZIGLMM Identified variance decomposition on a given cell type across both zero-inflated and continuous space using a zero-inflated general linear mixed model `glmmTMB`
Usage

```
varZIGLMM(
    TSAM_Object, 
    cellPopulation = NULL, 
    continuousRandom = NULL, 
    ziRandom = NULL, 
    zi_threshold = 0.1, 
    verbose = FALSE, 
    numCores = 1 
)
```

Arguments

- **TSAM_Object**: A SummarizedExperiment object generated from `getSampleTileMatrix`.
- **cellPopulation**: Name of a cell type(s), or 'all'. The function will combine the cell types mentioned into one matrix before running the model.
- **continuousRandom**: Random effects to test in the continuous portion. All factors must be found in column names of the TSAM_Object metadata, except for FragNumber and CellCount, which will be extracted from the TSAM_Object’s metadata.
- **ziRandom**: Random effects to test in the zero-inflated portion. All factors must be found in column names of the TSAM_Object colData metadata, except for FragNumber and CellCount, which will be extracted from the TSAM_Object’s metadata.
- **zi_threshold**: Zero-inflated threshold (range = 0-1), representing the fraction of samples with zeros. When the percentage of zeros in the tile is between 0 and zi_threshold, samples with zeroes are dropped and only the continuous formula is used. Use this parameter at your own risk. Default is 0.
- **verbose**: Set TRUE to display additional messages. Default is FALSE.
- **numCores**: integer. Number of cores to parallelize across.

Value

results a SummarizedExperiment containing results from ZIGLMM (Fixed effect estimates, P-values, and Std Error)

Examples

```r
## Not run:
modellist <- runZIGLMM(STM[c(1:1000), ], 
    cellPopulation = "CD16 Mono", 
    continuousRandom = c("Age", "Sex", "Days"), 
    ziRandom = c("FragNumber", "Days"), 
    verbose = TRUE, 
    numCores = 1
)
## End(Not run)
```

```r
## Not run:
modellist <- runZIGLMM(STM[c(1:1000), ], 
    cellPopulation = "CD16 Mono", 
    continuousRandom = c("Age", "Sex", "Days"), 
    ziRandom = c("FragNumber", "Days"), 
    verbose = TRUE, 
    numCores = 35
)
## End(Not run)
```
youden_threshold

___

youden_threshold youden_threshold

---

Description

Trained regression model for predicting a cutoff threshold for peak calling. Call: loess(formula = OptimalCutpoint ~ Ncells, data = thresh_df)

Usage

youden_threshold

Format

A list of 18 regression variables

Details

Number of Observations: 27 Equivalent Number of Parameters: 5.98 Residual Standard Error: 0.02121
Index

* datasets
  .counts_plot_default_theme, 3
  .gene_plot_theme, 4
  exampleBlackList, 13
  exampleCellColData, 14
  exampleFragments, 14
  finalModelObject, 22
  youden_threshold, 54
  .callOpenTiles_ArchR(callOpenTiles), 9
  .counts_plot_default_theme, 3
  .gene_plot_theme, 4

  addAccessibilityShift, 4
  addMotifSet, 5
  annotateTiles, 6

  bulkDimReduction, 7
  bulkUMAP, 8

  callOpenTiles, 9
  callOpenTiles,ArchRProject-method (callOpenTiles), 9
  callOpenTiles,GRangesList-method (callOpenTiles), 9
  callOpenTiles,list-method (callOpenTiles), 9
  combineSampleTileMatrix, 12

  differentialsToGRanges, 13
  exampleBlackList, 13
  exampleCellColData, 14
  exampleFragments, 14
  exportCoverage, 15
  exportDifferentials, 16
  exportMotifs, 17
  exportOpenTiles, 18
  exportSmoothedInsertions, 19
  extractRegion, 20

  filterCoAccessibleLinks, 21

  finalModelObject, 22
  getAltTSS, 23
  getAnnotationDbFromInstalledPkgname, 24
  getBSgenome, 11, 24
  getCellPopMatrix, 24
  getCellTypes, 25
  getCellTypeTiles, 25
  getCoAccessibleLinks, 26
  getCoverage, 27
  getDifferentialAccessibleTiles, 28
  getIntensityThreshold, 30
  getModelValues, 30
  getPopFrags, 31
  getPromoterGenes, 32
  getSampleCellTypeMetadata, 32
  getSampleTileMatrix, 33
  getSequencingBias, 34
  glmmTMB, 40, 48, 52
  GRangesToString, 35

  lm, 39, 47

  mergeTileResults, 36
  MotifEnrichment, 37
  MotifSetEnrichmentAnalysis, 37

  packMOCHA, 38, 52
  pbapply, 28
  pilotLMEM, 39
  pilotZIGLMM, 40
  plotConsensus, 41
  plotIntensityDistribution, 42
  plotRegion, 43

  renameCellTypes, 46
  runLMEM, 47
  runZIGLMM, 48

  StringsToGRanges, 49
subsetMOCHAObject, 50

testCoAccessibility, 51

umap, 8
unpackMOCHA, 38, 52, 52

varZIGLMM, 52

youden_threshold, 54