

# Package ‘MOCHA’

June 12, 2023

**Type** Package

**Title** Modeling for Single-Cell Open Chromatin Analysis

**Version** 1.0.0

**Maintainer** Imran McGrath <imran.mcgrath@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)

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

**Suggests** ArchR, RMariaDB, motifmatchr, BiocManager, TxDb.Hsapiens.UCSC.hg38.refGene, org.Hs.eg.db, BSgenome.Hsapiens.UCSC.hg19, withr, knitr, rmarkdown, chromVAR, testthat (>= 3.0.0), uwot, irlba, glmmTMB, Matrix, lmerTest

**Additional\_repositories** <https://imran-aifi.github.io/drat>

**biocViews** Data analysis, Chromatin Accessibility, Single cell, scATAC,  
Software, Visualization

**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** 2023-06-12 12:30:07 UTC

## R topics documented:

.counts_plot_default_theme . . . . .	3
.gene_plot_theme . . . . .	3
addAccessibilityShift . . . . .	4
addMotifSet . . . . .	4
annotateTiles . . . . .	5
bulkDimReduction . . . . .	6
bulkUMAP . . . . .	7
callOpenTiles . . . . .	8
combineSampleTileMatrix . . . . .	11
differentialsToGRanges . . . . .	12
exampleBlackList . . . . .	13
exampleCellColData . . . . .	13
exampleFragments . . . . .	13
extractRegion . . . . .	14
filterCoAccessibleLinks . . . . .	15
finalModelObject . . . . .	16
getAltTSS . . . . .	16
getCellPopMatrix . . . . .	17
getCoAccessibleLinks . . . . .	18
getCoverage . . . . .	19
getDifferentialAccessibleTiles . . . . .	20
getModelValues . . . . .	22
getPopFrag . . . . .	22
getSampleTileMatrix . . . . .	23
GRangesToString . . . . .	25
MotifEnrichment . . . . .	25
MotifSetEnrichmentAnalysis . . . . .	26
pilotLMEM . . . . .	27
pilotZIGLMM . . . . .	27
plotConsensus . . . . .	28

<code>.counts_plot_default_theme</code>	3
<code>plotRegion</code>	29
<code>runLMEM</code>	33
<code>runZIGLMM</code>	34
<code>StringsToGRanges</code>	35
<code>subsetMOCHAObject</code>	35
<code>testCoAccessibilityChromVar</code>	36
<code>testCoAccessibilityRandom</code>	37
<code>youden_threshold</code>	38
<b>Index</b>	<b>40</b>

---

`.counts_plot_default_theme`  
*Default ggplot theme for counts plot*

---

**Description**

Default ggplot theme for counts plot

**Usage**

`.counts_plot_default_theme`

**Format**

An object of class `list` of length 10.

---

`.gene_plot_theme`      *Common theme for gene plots*

---

**Description**

Common theme for gene plots

**Usage**

`.gene_plot_theme`

**Format**

An object of class `list` of length 5.

---

```
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.

### 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 PFMMatrix, PFMMatrixList, PWMMatrix, or PWMMatrixList
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 just the motifs from motifmatchr.
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

**Examples**

```
## 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)
```

---

 annotateTiles

 annotateTiles
 

---

**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/package>

**Usage**

```
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**

```
## Not run:
library(TxDb.Hsapiens.UCSC.hg38.refGene)
library(org.Hs.eg.db)
SampleTileMatricesAnnotated <- MOCHA::annotateTiles(
  SampleTileMatrices,
  TxDb = TxDb.Hsapiens.UCSC.hg38.refGene,
  Org = org.Hs.eg.db
)

## End(Not run)
```

---

bulkDimReduction	bulkDimReduction	
------------------	------------------	--

---

**Description**

`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/>

**Examples**

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

## End(Not run)
```

---

bulkUMAP

bulkUMAP

---

**Description**

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

**Usage**

```
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 <a href="#">umap</a> . 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 <a href="#">umap</a> .

**Value**

fullUMAP data.frame of UMAP values with metadata attached.

**Examples**

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

## End(Not run)
```

---

callOpenTiles	callOpenTiles <i>Perform peak-calling on a set of fragments or an ArchR Project.</i>
---------------	--

---

**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,
  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,  
  cellCol = "RG",  
  TxDb,  
  OrgDb,  
  outDir,  
  numCores = 30,  
  verbose = FALSE,  
  force = FALSE  
)  
  
## S4 method for signature 'list'  
callOpenTiles(  
  ATACFragments,  
  cellColData,  
  blackList,  
  genome,  
  cellPopLabel,  
  cellPopulations = "ALL",  
  studySignal = NULL,  
  cellCol = "RG",  
  TxDb,  
  OrgDb,  
  outDir,  
  numCores = 30,  
  verbose = FALSE,  
  force = FALSE  
)  
  
.callOpenTiles_ArchR(  
  ATACFragments,  
  cellPopLabel,  
  cellPopulations = "ALL",
```

```

    studySignal = NULL,
    TxDb,
    OrgDb,
    outDir = NULL,
    numCores = 30,
    verbose = FALSE,
    force = FALSE
)

```

## 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
genome	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 <a href="#">getBSgenome</a> .
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).
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 <a href="#">Bioconductor AnnotationData Packages</a> .
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 <a href="#">Bioconductor AnnotationData Packages</a>
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**

```
## Not run:
# Starting from an ArchR Project:
tileResults <- MOCHA::callOpenTiles(
  ArchRProj = myArchRProj,
  cellPopLabel = "celltype_labeling",
  cellPopulations = "CD4",
  TxDb = "TxDb.Hsapiens.UCSC.hg38.refGene",
  OrgDb = "org.Hs.eg.db",
  numCores = 1
)

## End(Not run)

# Starting from GRangesList
if (
  requireNamespace("BSgenome.Hsapiens.UCSC.hg19") &&
  requireNamespace("TxDb.Hsapiens.UCSC.hg38.refGene") &&
  requireNamespace("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",
    OrgDb = "org.Hs.eg.db",
    outDir = tempdir(),
    cellPopLabel = "Clusters",
    cellPopulations = c("C2", "C5"),
    numCores = 1
  )
}
```

---

combineSampleTileMatrix

combineSampleTileMatrix

---

**Description**

combineSampleTileMatrix combines all celltypes in a SampleTileMatrix object into a SummarizedExperiment with one single matrix across all cell types and samples, annotating GC bias using chromVAR.

**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.

**Value**

TileCorr A data.table correlation matrix

---

differentialsToGRanges

*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      *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/re>

**Usage**

```
exampleBlackList
```

**Format**

A GRanges object with 210 ranges and 2 metadata columns

---

exampleCellColData      *exampleCellColData*

---

**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/re>

**Usage**

```
exampleCellColData
```

**Format**

A DataFrame with 2217 rows and 3 columns

---

exampleFragments      *exampleFragments*

---

**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

---

extractRegion	extractRegion
---------------	---------------

---

**Description**

extractRegion will extract the coverage files created by callOpenTiles and return a specific region's coverage

**Usage**

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

**Arguments**

SampleTileObj	The SummarizedExperiment object output from getSampleTileMatrix
region	a GRanges object or vector or strings containing the regions of interest. Strings must be in the format "chr:start-end", e.g. "chr4:1300-2222".
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.
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 base-pairs, binning will be used. Default is 100000.
binSize	Optional, size of bins in basepairs when binning is used. Default is 250.

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**

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

---

```
filterCoAccessibleLinks
      filterCoAccessibleLinks
```

---

**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

**Value**

FilteredTileCorr The filtered correlation table with chr, start, and end site of each link

**Examples**

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

## End(Not run)
```

---

```
finalModelObject      finalModelObject
```

---

**Description**

Trained MOCHA models - LOESS and linear regression

**Usage**

```
finalModelObject
```

**Format**

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

**Loess** LOESS model

**Linear** Linear model

---

```
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**

```
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 be 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 <a href="#">Bioconductor AnnotationData Packages</a> .
OrgDb	The OrgDb-class genome wide annotation package for your organism (e.g. "org.Hs.eg.db"). This must be installed. See <a href="#">Bioconductor AnnotationData Packages</a>

**Value**

tpeaks A GRanges containing annotated peaks falling in TSS

---

```
getCellPopMatrix      getCellPopMatrix
```

---

**Description**

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

**Usage**

```
getCellPopMatrix(
  SampleTileObj,
  cellPopulation,
  dropSamples = TRUE,
  NtoZero = TRUE
)
```

**Arguments**

SampleTileObj	The output from <code>getSampleTileMatrix</code> , a <code>SummarizedExperiment</code> 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.

---

```
getCoAccessibleLinks  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**

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

**Arguments**

SampleTileObj	The <code>SummarizedExperiment</code> object output from <code>getSampleTileMatrix</code> containing your sample-tile matrices
cellPopulation	A string denoting the cell population of interest, which must be present in <code>SampleTileObj</code>
regions	a <code>GRanges</code> 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 <code>getDifferentialAccessibleTiles</code> .

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.
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:

Pimentel, Ronald Silva, "Kendall's Tau and Spearman's Rho for Zero-Inflated Data" (2009). Dissertations.

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

### Value

TileCorr A data.table correlation matrix

---

getCoverage	<i>Get sample-specific coverage files for each sample-cell population.</i>
-------------	--

---

### Description

getCoverage takes the output of MOCHA::getPopFragments and returns a GRanges of single-basepair resolution coverage.

### Usage

```
getCoverage(
  popFragments,
  normFactor,
  TxDb,
  cl,
  filterEmpty = FALSE,
  verbose = FALSE
)
```

**Arguments**

popFrag	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 <a href="#">Bioconductor AnnotationData Packages</a> .
cl	cl argument to <a href="#">pblapply</a>
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

---

```
getDifferentialAccessibleTiles
      getDifferentialAccessibleTiles
```

---

**Description**

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

**Usage**

```
getDifferentialAccessibleTiles(
  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 <code>getSampleTileMatrix</code>
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
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**

```
## 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)
```

---

getModelValues	<i>getModelValues from runZIGLMM output.</i>
----------------	--

---

**Description**

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)
```

**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**

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

---

getPopFragments	<i>Extract fragments by populations from an ArchR Project</i>
-----------------	---

---

**Description**

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

**Usage**

```
getPopFragments(  
  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.

---

```
getSampleTileMatrix  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

**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
  )
}
```



---

GRangesToString	GRangesToString <i>Converts a GRanges object to a string in the format 'chr1:100-200'</i>
-----------------	---

---

**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

---

MotifEnrichment	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

 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",
  numCores = 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

---

pilotLMEM                      *Execute a pilot run of single linear model on a subset of data*

---

### Description

pilotLMEM Runs linear mixed-effects modeling for continuous, non-zero inflated data using [lmer](#)

### Usage

```
pilotLMEM(
  ExperimentObj,
  cellPopulation = NULL,
  modelFormula = NULL,
  pilotIndices = 1:10,
  verbose = FALSE
)
```

### Arguments

**ExperimentObj**    A SummarizedExperiment object generated from `getSampleTileMatrix`, `chromVAR`, or other.

**cellPopulation**    A single cell population on which to run this pilot model

**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.

### Value

`modelList` a list of outputs from `lmerTest::lmer`

---

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

---

### Description

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**

```

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 <code>getSampleTileMatrix</code> , <code>chromVAR</code> , or other.
cellPopulation	A single cell population on which to run this pilot model
continuousFormula	The formula, see <a href="#">glmTMB</a> . Combined fixed and random effects formula, following lme4 syntax.
ziformula	The zero-inflated formula, see <a href="#">glmTMB</a> . a one-sided (i.e., no response variable) formula for zero-inflation combining fixed and random effects: the default <code>~0</code> specifies no zero-inflation. Specifying <code>~.</code> 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 <code>~.</code> 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 <code>zi_threshold</code> , 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.

**Value**

modelList a list of outputs from `glmTMB::glmTMB`

---

plotConsensus	plotConsensus
---------------	---------------

---

**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.

---

plotRegion	plotRegion
------------	------------

---

**Description**

plotRegion Plots the region that you've summarized across all cell groupings (groups=initial getPopFragments() 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 'whichGene' the **RMariaDB** package must be installed.

**Usage**

```
plotRegion(
  countSE,
  plotType = "area",
  base_size = 12,
```

```

counts_color = NULL,
range_label_size = 2,
legend.position = NULL,
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,
whichGene = NULL,
db_id_col = "REFSEQ",
collapseGenes = "None",
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,
  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.

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_y_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.
whichGene	Name of gene for plotting this specific gene in region.
db_id_col	Character value. Column in 'OrgDb' containing the output id for 'whichGene' 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.

<code>gene_theme_ls</code>	Named list of parameters passed to <code>'theme()'</code> for the gene plot. Default NULL will use <code>'.gene_plot_theme'</code>
<code>additionalGRangesTrack</code>	A <code>GRanges</code> object containing additional track plot data
<code>linkdf</code>	A dataframe with co-accessible links to display as an additional track
<code>showIdeogram</code>	Logical value, default TRUE. If TRUE plots the chromosome ideogram at the top of the multi-track plot
<code>ideogram_genome</code>	Character value, a genome name for the ideogram plot. Default <code>'hg19'</code> .
<code>relativeHeights</code>	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 = <code>c('Chr' = 0.9, 'Normalized Counts' = 7, 'Genes' = 2, 'AdditionalGRanges' = 4.5)</code>
<code>verbose</code>	Set TRUE to display additional messages. Default is FALSE.

**Value**

The input `ggplot` object with motif labels overlaid

**Examples**

```
## Not run:
# 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)
```



---

runLMEM	<i>Run Linear Mixed-Effects Modeling for continuous, non-zero inflated data</i>
---------	---

---

### Description

runLMEM Runs linear mixed-effects modeling for continuous, non-zero inflated data using [lmer](#)

### Usage

```
runLMEM(  
  ExperimentObj,  
  modelFormula = NULL,  
  initialSampling = 5,  
  verbose = FALSE,  
  numCores = 1  
)
```

### Arguments

ExperimentObj	A SummarizedExperiment object generated from <code>getSampleTileMatrix</code> , <code>chromVAR</code> , 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.
modelFormula	The formula to use with <code>lmerTest::lmer</code> , in the format ( <code>exp ~ factors</code> ). All factors must be found in column names of the ExperimentObj metadata. modelFormula must start with 'exp' as the response. See <a href="#">lmer</a> .
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

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

---

runZIGLMM	<i>Run Zero-inflated Generalized Linear Mixed Modeling on pseudobulked scATAC data</i>
-----------	--

---

## Description

runZIGLMM Runs linear mixed-effects modeling for zero-inflated data using [glmTMB](#).

## Usage

```
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 <code>getSampleTileMatrix</code> .
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 <code>glmTMB</code> . It should be in the format ( <code>exp ~ factors</code> ). All factors must be found in column names of the <code>TSAM_Object</code> metadata, except for <code>CellType</code> , <code>FragNumber</code> and <code>CellCount</code> , which will be extracted from the <code>TSAM_Object</code> . <code>modelFormula</code> must start with 'exp' as the response. See <a href="#">glmTMB</a> .
ziformula	The formula for the zero-inflated data that should be used within <code>glmTMB</code> . It should be in the format ( <code>~ factors</code> ). All factors must be found in column names of the <code>TSAM_Object</code> <code>colData</code> metadata, except for <code>CellType</code> , <code>FragNumber</code> and <code>CellCount</code> , which will be extracted from the <code>TSAM_Object</code> .
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 <code>zi_threshold</code> , 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**

returns a SummarizedExperiment containing LMEM results

**Examples**

```
## 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	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	subsetObject
-------------------	--------------

---

**Description**

subsetObject 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.

---

```
testCoAccessibilityChromVar
      testCoAccessibilityChromVar
```

---

**Description**

testCoAccessibilityChromVar takes an input set of tile pairs and tests whether they are significantly different compared to a background set found via ChromVAR

**Usage**

```
testCoAccessibilityChromVar(
  SampleTileObj,
  tile1,
  tile2,
  numCores = 1,
  ZI = TRUE,
  backNumber = 1000,
```

```

    returnBackground = FALSE,
    highMem = 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 ChromVAR-matched background pairs. Default is 1000.
returnBackground	Boolean, if TRUE return the background correlations as well as foreground. Default is FALSE.
highMem	Boolean to control memory usage. Default is FALSE. Only set highMem to TRUE if you have plenty of memory and want to run this function faster.
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

---

```

testCoAccessibilityRandom
      testCoAccessibilityRandom

```

---

### Description

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

**Usage**

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

**Arguments**

SampleTileObj	The SummarizedExperiment object output from <code>getSampleTileMatrix</code> 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

---

youden_threshold	<i>youden_threshold</i>
------------------	-------------------------

---

**Description**

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

*youden\_threshold*

39

**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, 3
- exampleBlackList, 13
- exampleCellColData, 13
- exampleFragments, 13
- finalModelObject, 16
- youden\_threshold, 38
- .callOpenTiles\_ArchR (callOpenTiles), 8
- .counts\_plot\_default\_theme, 3
- .gene\_plot\_theme, 3
- addAccessibilityShift, 4
- addMotifSet, 4
- annotateTiles, 5
- bulkDimReduction, 6
- bulkUMAP, 7
- callOpenTiles, 8
- callOpenTiles, ArchRProject-method (callOpenTiles), 8
- callOpenTiles, GRangesList-method (callOpenTiles), 8
- callOpenTiles, list-method (callOpenTiles), 8
- combineSampleTileMatrix, 11
- differentialsToGRanges, 12
- exampleBlackList, 13
- exampleCellColData, 13
- exampleFragments, 13
- extractRegion, 14
- filterCoAccessibleLinks, 15
- finalModelObject, 16
- getAltTSS, 16
- getBSgenome, 10
- getCellPopMatrix, 17

- getCoAccessibleLinks, 18
- getCoverage, 19
- getDifferentialAccessibleTiles, 20
- getModelValues, 22
- getPopFrag, 22
- getSampleTileMatrix, 23
- glmmTMB, 27, 28, 34
- GRangesToString, 25
- lmer, 27, 33
- MotifEnrichment, 25
- MotifSetEnrichmentAnalysis, 26
- pblapply, 20
- pilotLMEM, 27
- pilotZIGLMM, 27
- plotConsensus, 28
- plotRegion, 29
- runLMEM, 33
- runZIGLMM, 34
- StringsToGRanges, 35
- subsetMOCHAObject, 35
- testCoAccessibilityChromVar, 36
- testCoAccessibilityRandom, 37
- umap, 8
- youden\_threshold, 38