Package 'SpotClean'

December 6, 2024

Version 1.8.0

Date 2024/06/08

Title SpotClean adjusts for spot swapping in spatial transcriptomics data

Depends R (>= 4.2.0),

Imports stats, methods, utils, dplyr, S4Vectors, SummarizedExperiment, SpatialExperiment, Matrix, rhdf5, ggplot2, grid, readbitmap, rjson, tibble, viridis, grDevices, RColorBrewer, Seurat, rlang

Suggests testthat (>= 2.1.0), knitr, BiocStyle, rmarkdown, R.utils, spelling

biocViews DataImport, RNASeq, Sequencing, GeneExpression, Spatial, SingleCell, Transcriptomics, Preprocessing

Description SpotClean is a computational method to adjust for spot swapping in spatial transcriptomics data. Recent spatial transcriptomics experiments utilize slides containing thousands of spots with spot-specific barcodes that bind mRNA. Ideally, unique molecular identifiers at a spot measure spot-specific expression, but this is often not the case due to bleed from nearby spots, an artifact we refer to as spot swapping. SpotClean is able to estimate the contamination rate in observed data and decontaminate the spot swapping effect, thus increase the sensitivity and precision of downstream analyses.

License GPL-3

NeedsCompilation yes

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.1

URL https://github.com/zijianni/SpotClean

BugReports https://github.com/zijianni/SpotClean/issues

Language en-US

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

git_branch RELEASE_3_20

git_last_commit b4fab33

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2024-12-05

Author Zijian Ni [aut, cre] (<https://orcid.org/0000-0003-1181-8337>),
Christina Kendziorski [ctb]

Maintainer Zijian Ni <zni25@wisc.edu>

Contents

`SpotClean-package` *SpotClean: a computational method to adjust for spot swapping in spatial transcriptomics data*

Description

SpotClean is a computational method to adjust for spot swapping in spatial transcriptomics data. Recent spatial transcriptomics experiments utilize slides containing thousands of spots with spot-specific barcodes that bind mRNA. Ideally, unique molecular identifiers at a spot measure spot-specific expression, but this is often not the case due to bleed from nearby spots, an artifact we refer to as spot swapping. SpotClean is able to estimate the contamination rate in observed data and decontaminate the spot swapping effect, thus increase the sensitivity and precision of downstream analyses.

Details

To learn more about `SpotClean`, see the vignette using `browseVignettes(package = "SpotClean")`.

Author(s)

Maintainer: Zijian Ni <zni25@wisc.edu> (ORCID)

Other contributors:

- Christina Kendziorski [contributor]

See Also

Useful links:

- https://github.com/zijianni/SpotClean
- Report bugs at https://github.com/zijianni/SpotClean/issues

`arcScore` *Calculate the ambient RNA contamination (ARC) score*

Description

The ARC score is intended for subjectively measuring the level of contamination caused by ambient RNAs for a given spatial transcriptomics or droplet-based single-cell RNA-seq data. Intuitively, this score is a lower bound of the average proportion of contaminated expressions in tissue spots or cell droplets in the observed contaminated data.

ARC score is calculated as follows: (1) estimate the average amount of contaminated UMI counts received per spot (droplet) using the total UMI counts in background spots (empty droplets) divided by total number of spots (droplets). This is an underestimation as it does not account for contamination inside tissue spots or cell droplets. (2) estimate the average amount of observed UMI counts per tissue spot (cell droplet) using the total UMI counts in tissue spots (cell droplets) divided by number of tissue spots (cell droplets). (3) Divide the value in (1) by the value in (2) to get the ARC score.

This lower bound is very conservative as it only accounts for observable contamination in background spots or empty droplets, neglecting contamination in tissue spots or cell droplets. However, it is totally subjective, not relying on any complicated assumptions and modelling. ARC score serves as a relative contamination measurement for comparison among different datasets and protocols.

Usage

```
arcScore(object, ...)

## Default S3 method:
arcScore(object, background_bcs, ...)

## S3 method for class 'SummarizedExperiment'
arcScore(object, ...)
```

Arguments

`object`	A gene-by-barcode count matrix or a slide object created or inherited from `createSlide()`. Ideally this should give the raw count matrix with all genes and barcodes without any filtering.
`...`	Arguments passed to other methods
`background_bcs`	(vector of chr) Background barcodes in `count_mat`. For spatial data, these are background spots not covered by tissue. For single-cell data, these are empty droplets not containing cells.

Value

(num) The ARC score of given data.

Examples

```
data(mbrain_raw)
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                     "tissue_positions_list.csv"),
            tissue_img_file = file.path(spatial_dir,
                                     "tissue_lowres_image.png"),
            scale_factor_file = file.path(spatial_dir,
                                     "scalefactors_json.json"))

background_bcs <- dplyr::filter(mbrain_slide_info$slide, tissue==0)$barcode
arcScore(mbrain_raw, background_bcs)

mbrain_obj <- createSlide(mbrain_raw, mbrain_slide_info)
arcScore(mbrain_obj)
```

`convertToSeurat` *Convert slide object to Seurat object*

Description

This function converts our slide object of class `SummarizedExperiment` to Seurat object of class `Seurat` so that users can directly proceed with Seurat spatial analyses pipelines. Built based on Seurat's `Load10X_Spatial()`.

Usage

```
convertToSeurat(slide_obj, image_dir, slice = "slice1", filter_matrix = TRUE)
```

Arguments

`slide_obj`	A slide object created or inherited from `createSlide()`.
`image_dir`	(chr) Path to directory with 10X Genomics visium image data; should include files `tissue_lowres_image.png`, `scalefactors_json.json` and `tissue_positions_list.csv`.
`slice`	(chr) Name for the stored image of the tissue slice. Default: "slice1"
`filter_matrix`	(logical) If `TRUE`, only keep spots that have been determined to be over tissue. If `slide_obj` only contains tissue spots, `filter_matrix` has to be set `TRUE`. If `slide_obj` contains both tissue and background spots, setting `filter_matrix=TRUE` will subset the expression matrix to tissue spots only. Default: `TRUE`

Value

A Seurat object with spatial information.

Examples

```
# load count matrix and slide metadata
data(mbrain_raw)
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                     "tissue_positions_list.csv"),
            tissue_img_file = file.path(spatial_dir,
                                     "tissue_lowres_image.png"),
            scale_factor_file = file.path(spatial_dir,
                                     "scalefactors_json.json"))

# Create slide object
mbrain_obj <- createSlide(mbrain_raw,
                          mbrain_slide_info)

# Convert to Seurat object
seurat_obj <- convertToSeurat(mbrain_obj, spatial_dir, "raw")
str(seurat_obj)
```

---

createSlide *Create a new slide object*

Description

This function takes input of the count matrix (from `read10xRaw` or `read10xRawH5`) and slide information (from `read10xSlide` or manually specified data frame) and outputs a `SummarizedExperiment` object as our slide object for downstream decontamination and visualization.

Usage

```
createSlide(count_mat, slide_info, gene_cutoff = 0.1, verbose = TRUE)
```

Arguments

`count_mat`	(matrix of num) The raw gene-by-barcode count matrix. Can be either standard matrix format or sparse matrix format.
`slide_info`	(list or data.frame) A list of slide information from `read10xSlide()`, or a data frame only containing spot information like barcode, tissue, imagerow, imagecol, etc.
`gene_cutoff`	(num) Filter out genes with average expressions among tissue spots below or equal to this cutoff. Default: 0.1
`verbose`	(logical) Whether print progress information. Default: `TRUE`

Value

A `SummarizedExperiment` object containing gene expression and spot metadata.

Examples

```
data(mbrain_raw)
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                     "tissue_positions_list.csv"),
             tissue_img_file = file.path(spatial_dir,
                                     "tissue_lowres_image.png"),
             scale_factor_file = file.path(spatial_dir,
                                     "scalefactors_json.json"))

mbrain_obj <- createSlide(mbrain_raw,
                          mbrain_slide_info)
mbrain_obj
```

---

keepHighGene *Filter and return highly expressed or highly variable genes*

Description

This function is for filtering genes in the expression count matrix based on their average expression and variability. Usually genes with low expressions and variability are less interesting and do not contribute too much to downstream analyses, but rather bring technical noise. It is always recommended to pre-filter gene expression matrix before any analyses.

We apply the function `FindVariableFeatures` with `selection.method = "mvp"` from package `Seurat` on log transformed expression matrix to detect high variable genes. This method does not require a pre-specified number of high variable genes.

Usage

```
keepHighGene(
  count_mat,
  top_high = 5000,
  mean_cutoff = 1,
  return_matrix = FALSE,
  verbose = TRUE
)
```

Arguments

`count_mat`	(matrix of num) Input count matrix to be filtered. Can be either standard matrix format or sparse matrix format.
`top_high`	(int) Only look for highly expressed and variable genes within this number of top expressed genes. Default: 5000.
`mean_cutoff`	(num) Genes with average expressions among all spots exceeding this cutoff are kept as highly expressed genes.
`return_matrix`	(logical) Whether return filtered matrix instead of gene names. Default: `FALSE`
`verbose`	(logical) Whether print progress information. Default: `TRUE`

Value

A vector of gene names or a filtered expression count matrix with the same class as `count_mat`.

Examples

```
data(mbrain_raw)
dim(mbrain_raw)

mbrain_raw_f <- keepHighGene(mbrain_raw, mean_cutoff=100,
                             return_matrix=TRUE)
dim(mbrain_raw_f)
```

`mbrain_raw` *Example 10x Visium spatial data: raw count matrix*

Description

This dataset contains one sparse count matrix `mbrain_raw` of gene expressions. The original dataset can be found at https://support.10xgenomics.com/spatial-gene-expression/datasets/1.0.0/V1_Adult_Mouse_Brain. For simplicity, we only keep the top 100 highest expressed genes in this example data.

If users read raw data using `read10xRaw` (or `read10xRawH5`), they should get exactly the same format as the object in this dataset.

Usage

```
data(mbrain_raw)
```

Format

An object of class `"dgCMatrix"`.

Source

V1_Adult_Mouse_Brain

Examples

```
data(mbrain_raw)
str(mbrain_raw)
```

`read10xRaw` *Read 10x Space Ranger output data*

Description

`read10xRaw()` is a one-line handy function for reading the raw expression data from 10x Space Ranger outputs and producing a count matrix as an R object.

`read10xRawH5()` is for reading 10x Space Ranger output HDF5 file (ended with .h5).

`read10xSlide()` is for reading slide information (e.g. spot positions) and the tissue image from 10x Space Ranger outputs. This function is developed based on 10x's secondary analysis pipeline https://support.10xgenomics.com/spatial-gene-expression/software/pipelines/latest/rkit.

Usage

```
read10xRaw(count_dir = NULL, row_name = "symbol", meta = FALSE)

read10xRawH5(h5_file, row_name = "symbol", meta = FALSE)

read10xSlide(tissue_csv_file, tissue_img_file = NULL, scale_factor_file = NULL)
```

Arguments

`count_dir`	(chr) The directory of 10x output matrix data. The directory should include three files: barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz.
`row_name`	(chr) Specify either using gene symbols (`row_name = "symbol"`) or gene Ensembl IDs (`row_name = "id"`) as row names of the count matrix. Default: `row_name = "symbol"`
`meta`	(logical) If `TRUE`, `read10xRaw` or `read10xRawH5` returns a list containing both the count matrix and metadata of genes (features). Metadata includes feature names, IDs and other additional information depending on Space Ranger output. If `FALSE`, only returns the count matrix. Default: `FALSE`
`h5_file`	(chr) The path of 10x output matrix HDF5 file (ended with .h5).
`tissue_csv_file`	(chr) The path of 10x output CSV file of spot positions, usually named `tissue_positions_list.csv` for Space Ranger V1 and `tissue_positions.csv` for Space Ranger V2.
`tissue_img_file`	(chr) The path of the 10x output low resolution tissue image in PNG format, usually named `tissue_lowres_image.png`. If `NULL`, the returned slide data does not contain image information. Please do provide this file if you could find it. Default: `NULL`
`scale_factor_file`	(chr) The path of the 10x output scale factor file in json format, usually named `scalefactors_json.json`. If `NULL`, spot positions in image will not be corrected by the scale factor. Please do provide this file if you could find it. Default: `NULL`

Value

If `meta = TRUE`, `read10xRaw()` or `read10xRawH5()` returns a list of two elements: a "dgCMatrix" sparse matrix containing expression counts and a data frame containing metadata of genes (features). For the count matrix, each row is a gene (feature) and each column is a spot barcode. If `meta = FALSE`, only returns the count matrix.

`read10xSlide()` returns a list of two objects. The first object, slide, is a data.frame where each row corresponds to a spot and each column corresponds to slide information such as row and column positions on the slide. The second object, grob, is a Grid Graphical Object of the tissue image when specifying `tissue_img_file`.

Examples

```
# simulate 10x output files of count matrix
data(mbrain_raw)
data_dir <- file.path(tempdir(),"sim_example")
dir.create(data_dir)
matrix_dir <- file.path(data_dir,"matrix.mtx")
barcode_dir <- gzfile(file.path(data_dir, "barcodes.tsv.gz"), open="wb")
gene_dir <- gzfile(file.path(data_dir, "features.tsv.gz"), open="wb")

# For simplicity, use gene names to generate gene IDs to fit the format.
gene_name <- rownames(mbrain_raw)
gene_id <- paste0("ENSG_fake_",gene_name)
barcode_id <- colnames(mbrain_raw)

Matrix::writeMM(mbrain_raw,file = matrix_dir)
write(barcode_id,file = barcode_dir)
write.table(cbind(gene_id,gene_name,"type"),file = gene_dir,
    sep = "\t", quote = FALSE, col.names = FALSE, row.names = FALSE)
R.utils::gzip(matrix_dir)
close(barcode_dir)
close(gene_dir)


# read expression count matrix
list.files(data_dir)
mbrain_raw_new <- read10xRaw(data_dir)
str(mbrain_raw_new)
identical(mbrain_raw, mbrain_raw_new)

# read slide metadata
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
list.files(spatial_dir)
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                     "tissue_positions_list.csv"),
             tissue_img_file = file.path(spatial_dir,
                                     "tissue_lowres_image.png"),
             scale_factor_file = file.path(spatial_dir,
                                     "scalefactors_json.json"))
str(mbrain_slide_info)
```

`spotclean` *Decontaminate spot swapping effect in spatial transcriptomics data*

Description

This is the main function implementing the SpotClean method for decontaminating spot swapping effect in spatial transcriptomics data.

Usage

```
spotclean(slide_obj, ...)

## S3 method for class 'SummarizedExperiment'
spotclean(
  slide_obj,
  gene_keep = NULL,
  maxit = 30,
  tol = 1,
  candidate_radius = 5 * seq_len(6),
  kernel = "gaussian",
  verbose = TRUE,
  ...
)

## S3 method for class 'SpatialExperiment'
spotclean(
  slide_obj,
  gene_keep = NULL,
  gene_cutoff = 0.1,
  maxit = 30,
  tol = 1,
  candidate_radius = 5 * seq_len(6),
  kernel = "gaussian",
  verbose = TRUE,
  ...
)
```

Arguments

`slide_obj`	A slide object created or inherited from `createSlide()`, or a `SpatialExperiment` object created from `SpatialExperiment::read10xVisium()`.
`...`	Arguments passed to other methods
`gene_keep`	(vector of chr) Gene names to keep for decontamination. We recommend not decontaminating lowly expressed and lowly variable genes in order to save computation time. Even if user include them, their decontaminated expressions will not change too much from raw expressions. When setting to `NULL`, `keepHighGene()` will be automatically called to filter out lowly expressed and lowly variable genes before decontamination. Default: `NULL`.
`maxit`	(int) Maximum iteration for EM parameter updates. Default: 30.

`tol` (num) Tolerance to define convergence in EM parameter updates. When the element-wise maximum difference between current and updated parameter matrix is less than `tol`, parameters are considered converged. Default: 1

`candidate_radius` (vector of num) Candidate contamination radius. A series of radius to try when estimating contamination parameters. Default: c(5, 10, 15, 20, 25, 30)

`kernel` (chr): name of kernel to use to model local contamination. Supports "gaussian", "linear", "laplace", "cauchy". Default: "gaussian".

`verbose` (logical) Whether print progress information. Default: `TRUE`

`gene_cutoff` (num) Filter out genes with average expressions among tissue spots below or equal to this cutoff. Only applies to `SpatialExperiment` object. Default: 0.1.

Details

Briefly, the contamination level for the slide is estimated based on the total counts of all spots. UMI counts travelling around the slide are assumed to follow Poisson distributions and modeled by a mixture of Gaussian (proximal) and uniform (distal) kernels. The underlying uncontaminated gene expressions are estimated by EM algorithm to maximize the data likelihood. Detailed derivation can be found in our manuscript.

Value

For slide object created from `createSlide()`, returns a slide object where the decontaminated expression matrix is in the "decont" assay slot and the contamination statistics are in metadata slots. Contamination statistics include ambient RNA contamination (ARC) score, bleeding rate, distal rate, contamination radius, contamination kernel weight matrix, log-likelihood value in each iteration, estimated proportion of contamination in each tissue spot in observed data. Since decontaminated and raw data have different number of columns, they can not be stored in a single object.

For `SpatialExperiment` object created from `SpatialExperiment::read10xVisium()`, returns a `SpatialExperiment` object where the decontaminated expression matrix is in the "decont" assay slot and the contamination statistics are in metadata slots. Raw expression matrix is also stored in the "counts" assay slot. Genes are filtered based on `gene_cutoff`.

Examples

```
data(mbrain_raw)
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                    "tissue_positions_list.csv"),
             tissue_img_file = file.path(spatial_dir,
                                    "tissue_lowres_image.png"),
             scale_factor_file = file.path(spatial_dir,
                                    "scalefactors_json.json"))
mbrain_obj <- createSlide(mbrain_raw,
                          mbrain_slide_info)

mbrain_decont_obj <- spotclean(mbrain_obj, tol=10, candidate_radius=20)
mbrain_decont_obj
```

`visualizeHeatmap` *Visualize spot values on the 2D slide*

Description

Generate and visualize spot labels on the 2D slide as a ggplot2 object. Spot labels can be given via parameter `label` as external input or one column in the `slide` slot of the slide object's metadata via `label_col`. Exactly one of `label` and `label_col` must be specified.

Usage

```
visualizeHeatmap(object, ...)

## Default S3 method:
visualizeHeatmap(
  object,
  value,
  exp_matrix = NULL,
  subset_barcodes = NULL,
  logged = TRUE,
  viridis = TRUE,
  legend_range = NULL,
  title = "",
  legend_title = NULL,
  ...
)

## S3 method for class 'SummarizedExperiment'
visualizeHeatmap(
  object,
  value,
  subset_barcodes = NULL,
  logged = TRUE,
  viridis = TRUE,
  legend_range = NULL,
  title = "",
  legend_title = NULL,
  ...
)
```

Arguments

`object`	A slide object created or inherited from `createSlide()`, or a `data.frame` of slide information with columns: barcodes, tissue, imagerow, imagecol, etc.
`...`	Arguments passed to other methods
`value`	(numeric vector or chr) Either a vector of numeric values for all spots, or the name of one gene in `exp_matrix` or in the expression matrix in the slide object. In the former case, the order of values in the vector should match the spot barcodes in the slide object.

exp_matrix (matrix of num) When `object` is a data frame and `value` is a single character string, will search the matching gene in `exp_matrix` and plot the gene expression. Default: `NULL`

subset_barcodes (vector of chr) A subset of spot barcodes to plot. By default it plots all spots in the slide object. This can be useful when only plotting tissue spots or specific tissue types or regions. Default: `NULL`

logged (logical) Specify if the color scale is log1p transformed. Default: `TRUE`

viridis (logical) If true, color scale uses viridis. Otherwise, use rainbow. Default: `TRUE`

legend_range (length 2 vector of num) Custom legend range of the value. By default uses the range of the plotted values. Default: `NULL`

title (chr) Title of the plot. Default: `""`

legend_title (chr) Title of the legend. Under default, use `value` as legend title. Default: `NULL`

Value

A `ggplot2` object.

Examples

```
data(mbrain_raw)
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                     "tissue_positions_list.csv"),
             tissue_img_file = file.path(spatial_dir,
                                     "tissue_lowres_image.png"),
             scale_factor_file = file.path(spatial_dir,
                                     "scalefactors_json.json"))
mbrain_obj <- createSlide(mbrain_raw,
                          mbrain_slide_info)

gp <- visualizeHeatmap(mbrain_obj, "Bc1",
                       title="mbrain", legend_title="Bc1 expression")
plot(gp)
```

---

visualizeLabel *Visualize spot labels on the 2D slide*

Description

Generate and visualize spot labels on the 2D slide as a ggplot2 object. Spot labels can be given via parameter `label` as external input or one column in the `slide` slot of the slide object's metadata via `label_col`. Exactly one of `label` and `label_col` must be specified.

Usage

```
visualizeLabel(object, ...)

## Default S3 method:
visualizeLabel(
  object,
  label = "tissue",
  subset_barcodes = NULL,
  title = "",
  legend_title = NULL,
  ...
)

## S3 method for class 'SummarizedExperiment'
visualizeLabel(
  object,
  label = "tissue",
  subset_barcodes = NULL,
  title = "",
  legend_title = NULL,
  ...
)
```

Arguments

object	A slide object created or inherited from `createSlide()`, or a `data.frame` of slide information with columns: barcodes, tissue, imagerow, imagecol, etc.
...	Arguments passed to other methods
label	(categorical vector or chr) Either a vector of labels for all spots, or the column name in the `slide` slot of the slide object's metadata. In the former case, the order of values in the vector should match the spot barcodes in the slide object. In the latter case, The column should be categorical with reasonably number of unique labels. Default: `"tissue"`
subset_barcodes	(vector of chr) A subset of spot barcodes to plot. By default it plots all spots in the slide object. This can be useful when only plotting tissue spots or specific tissue types or regions. Default: `NULL`
title	(chr) Title of the plot. Default: `""`
legend_title	(chr) Title of the legend. Under default, use `label` as legend title. Default: `NULL`

Value

A `ggplot2` object.

Examples

```
data(mbrain_raw)
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                    "tissue_positions_list.csv"),
```

```
            tissue_img_file = file.path(spatial_dir,
                                        "tissue_lowres_image.png"),
            scale_factor_file = file.path(spatial_dir,
                                        "scalefactors_json.json"))
mbrain_obj <- createSlide(mbrain_raw,
                          mbrain_slide_info)
gp <- visualizeLabel(mbrain_obj, label="tissue",
                     title="mbrain", legend_title="tissue or background")
plot(gp)
```

`visualizeSlide` *Visualize the Visium slide image*

Description

Generate and visualize the tissue image as a ggplot2 object. Users can manually add and modify layers (e.g. title, axis) following ggplot2's syntax.

Usage

```
visualizeSlide(slide_obj, title = "")
```

Arguments

`slide_obj`	A slide object created or inherited from `createSlide()`.
`title`	(chr) Title of the plot. Default: `""`

Value

A `ggplot2` object.

Examples

```
data(mbrain_raw)
spatial_dir <- system.file(file.path("extdata",
                                     "V1_Adult_Mouse_Brain_spatial"),
                           package = "SpotClean")
mbrain_slide_info <- read10xSlide(tissue_csv_file=file.path(spatial_dir,
                                        "tissue_positions_list.csv"),
            tissue_img_file = file.path(spatial_dir,
                                        "tissue_lowres_image.png"),
            scale_factor_file = file.path(spatial_dir,
                                        "scalefactors_json.json"))
mbrain_obj <- createSlide(mbrain_raw,
                          mbrain_slide_info)
gp <- visualizeSlide(mbrain_obj)
plot(gp)
```

Index