Package: spatialHeatmap (via r-universe)

November 1, 2026

Type Package

Title spatialHeatmap: Visualizing Spatial Assays in Anatomical Images and Large-Scale Data Extensions

Version 2.18.2

Date 2026-05-12

Description The spatialHeatmap package offers the primary functionality for visualizing cell-, tissue- and organ-specific assay data in spatial anatomical images. Additionally, it provides extended functionalities for large-scale data mining routines and co-visualizing bulk and single-cell data. A description of the project is available here: https://spatialheatmap.org.

License Artistic-2.0

Encoding UTF-8

biocViews Spatial, Visualization, Microarray, Sequencing, GeneExpression, DataRepresentation, Network, Clustering, GraphAndNetwork, CellBasedAssays, ATACSeq, DNASeq, TissueMicroarray, SingleCell, CellBiology, GeneTarget

VignetteBuilder knitr

Depends R (>= 4.1.0)

Imports data.table, dplyr, edgeR, genefilter, ggplot2, grImport, grid, gridExtra, igraph, methods, Matrix, rsvg, shiny, grDevices, graphics, ggplotify, parallel, reshape2, stats, SummarizedExperiment, SingleCellExperiment, shinydashboard, S4Vectors, spsComps (>= 0.3.3.0), tibble, utils, xml2

Suggests AnnotationDbi, av, BiocParallel, BiocFileCache, BiocGenerics, BiocStyle, BiocSingular, Biobase, bluster, cachem, DESeq2, dendextend, DT, dynamicTreeCut, DelayedArray, flashClust, gplots, ggdendro, HDF5Array, htmltools, htmlwidgets, jsonlite, kableExtra, knitr, limma, magick, memoise, ExpressionAtlas, GEOquery, org.Hs.eg.db, org.Mm.eg.db, org.At.tair.db, org.Dr.eg.db, org.Dm.eg.db, pROC, plotly, rmarkdown, rappdirs, RUnit, Rtsne, rhdf5, scater, scrapper, shinyWidgets, shinyjs,

shinyBS, sortable, Seurat, sparkline, spsUtil, uwot, UpSetR, visNetwork, WGCNA, xlsx, yaml

BugReports https://github.com/jianhaizhang/spatialHeatmap/issues

URL https://spatialheatmap.org, https://github.com/jianhaizhang/spatialHeatmap

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libpng-dev librsvg2-dev libuv1-dev libxml2-dev libssl-dev zlib1g-dev

Repository Bioconductor 3.23

Date/Publication 2026-05-13 03:58:24 UTC

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

RemoteRef RELEASE_3_23

RemoteSha 185f8256ede039fde5f5d6b47ae893e672bc71c7

Contents

spatialHeatmap-package . . . 3
adj_mod . . . 7
aggr_rep . . . 11
aSVG.remote.repo . . . 14
cell_group . . . 15
cluster_cell . . . 16
coclus_opt . . . 18
cocluster . . . 20
com_factor . . . 26
covis . . . 27
custom_shiny . . . 36
cut_dendro . . . 40
cvt_id . . . 41
data_ref . . . 42
database . . . 43
edit_tar . . . 46
filter_data . . . 47
matrix_hm . . . 51
network . . . 55
norm_cell . . . 59
norm_data . . . 61
norm_srsc . . . 64
opt_bar . . . 66
opt_setting . . . 67
opt_violin . . . 68
optimal_k . . . 69
plot_dim . . . 70
plot_kmeans . . . 72

plot_meta . . . 73
process_cell_meta . . . 81
qc_cell . . . 83
read_cache . . . 84
read_fr . . . 85
read_svg . . . 86
reduce_dim . . . 87
reduce_rep . . . 89
return_feature . . . 90
save_cache . . . 92
shiny_shm . . . 93
shm . . . 96
spatial_hm . . . 103
SpatialEnrichment . . . 112
SPHM-class . . . 118
SPHMMethods . . . 119
submatrix . . . 121
SVG-class . . . 125
SVGMethods . . . 127
true_bulk . . . 130
update_feature . . . 131
write_svg . . . 132

Index **134**

`spatialHeatmap-package`

spatialHeatmap: Visualizing Spatial Assays in Anatomical Images and Large-Scale Data Extensions Spatial Heatmap, Spatial Enrichment, Data Mining, Co-visualization

---

Description

The spatialHeatmap package offers the primary functionality for visualizing cell-, tissue- and organ-specific assay data in spatial anatomical images. Additionally, it provides extended functionalities for large-scale data mining routines and co-visualizing bulk and single-cell data. A description of the project is available here: https://spatialheatmap.org.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

The spatialHeatmap package provides functionalities for visualizing cell-, tissue- and organ-specific data of biological assays by coloring the corresponding spatial features defined in anatomical images according to a numeric color key. The color scheme used to represent the assay values can

be customized by the user. This core functionality is called a spatial heatmap plot. It is enhanced with nearest neighbor visualization tools for groups of measured items (e.g. gene modules) sharing related abundance profiles, including matrix heatmaps combined with hierarchical clustering dendrograms and network representations. The functionalities of spatialHeatmap can be used either in a command-driven mode from within R or a graphical user interface (GUI) provided by a Shiny App that is also part of this package. While the R-based mode provides flexibility to customize and automate analysis routines, the Shiny App includes a variety of convenience features that will appeal to many biologists. Moreover, the Shiny App has been designed to work on both local computers as well as server-based deployments (e.g. cloud-based or custom servers) that can be accessed remotely as a centralized web service for using spatialHeatmap's functionalities with community and/or private data.

As anatomical images the package supports both tissue maps from public repositories and custom images provided by the user. In general any type of image can be used as long as it can be provided in SVG (Scalable Vector Graphics) format, where the corresponding spatial features have been defined (see aSVG below). The numeric values plotted onto a spatial heatmap are usually quantitative measurements from a wide range of profiling technologies, such as microarrays, next generation sequencing (e.g. RNA-Seq and scRNA-Seq), proteomics, metabolomics, or many other small- or large-scale experiments. For convenience, several preprocessing and normalization methods for the most common use cases are included that support raw and/or preprocessed data. Currently, the main application domains of the spatialHeatmap package are numeric data sets and spatially mapped images from biological and biomedical areas. Moreover, the package has been designed to also work with many other spatial data types, such a population data plotted onto geographic maps. This high level of flexibility is one of the unique features of spatialHeatmap. Related software tools for biological applications in this field are largely based on pure web applications (Winter et al. 2007; Waese et al. 2017) or local tools (Maag 2018; Muschelli, Sweeney, and Crainiceanu 2014) that typically lack customization functionalities. These restrictions limit users to utilizing pre-existing expression data and/or fixed sets of anatomical image collections. To close this gap for biological use cases, we have developed spatialHeatmap as a generic R/Bioconductor package for plotting quantitative values onto any type of spatially mapped images in a programmable environment and/or in an intuitive to use GUI application.

Author(s)

Jianhai Zhang [aut, trl, cre], Le Zhang [aut], Jordan Hayes [aut], Brendan Gongol [aut], Alexander Borowsky [aut], Julia Bailey-Serres [aut], Thomas Girke [aut] Author: Jianhai Zhang [aut, trl, cre], Le Zhang [aut], Jordan Hayes [aut], Brendan Gongol [aut], Alexander Borowsky [aut], Julia Bailey-Serres [aut], Thomas Girke [aut] Jianhai Zhang (PhD candidate at Genetics, Genomics and Bioinformatics, University of California, Riverside), Dr. Thomas Girke (Professor at Department of Botany and Plant Sciences, University of California, Riverside) Maintainer: Jianhai Zhang <jzhan067@ucr.edu> Jianhai Zhang <jzhan067@@ucr.edu>.

References

https://www.w3schools.com/graphics/svg_intro.asp

https://shiny.rstudio.com/tutorial/

https://shiny.rstudio.com/articles/datatables.html

https://rstudio.github.io/DT/010-style.html

https://plot.ly/r/heatmaps/

https://www.gimp.org/tutorials/

https://inkscape.org/en/doc/tutorials/advanced/tutorial-advanced.en.html

http://www.microugly.com/inkscape-quickguide/

https://cran.r-project.org/web/packages/visNetwork/vignettes/Introduction-to-visNetwork.html

https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg

Winter, Debbie, Ben Vinegar, Hardeep Nahal, Ron Ammar, Greg V Wilson, and Nicholas J Provart. 2007. "An 'Electronic Fluorescent Pictograph' Browser for Exploring and Analyzing Large-Scale Biological Data Sets." PLoS One 2 (8): e718

Waese, Jamie, Jim Fan, Asher Pasha, Hans Yu, Geoffrey Fucile, Ruian Shi, Matthew Cumming, et al. 2017. "EPlant: Visualizing and Exploring Multiple Levels of Data for Hypothesis Generation in Plant Biology." Plant Cell 29 (8): 1806–21

Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angelica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9

Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas

Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8

McCarthy, Davis J., Chen, Yunshun, Smyth, and Gordon K. 2012. "Differential Expression Analysis of Multifactor RNA-Seq Experiments with Respect to Biological Variation." Nucleic Acids Research 40 (10): 4288–97

Maag, Jesper L V. 2018. "Gganatogram: An R Package for Modular Visualisation of Anatograms and Tissues Based on Ggplot2." F1000Res. 7 (September): 1576

Muschelli, John, Elizabeth Sweeney, and Ciprian Crainiceanu. 2014. "BrainR: Interactive 3 and 4D Images of High Resolution Neuroimage Data." R J. 6 (1): 41–48

Morgan, Martin, Valerie Obenchain, Jim Hester, and Hervé Pagès. 2018. SummarizedExperiment: SummarizedExperiment Container

Winston Chang, Joe Cheng, JJ Allaire, Yihui Xie and Jonathan McPherson (2017). shiny: Web Application Framework for R. R package version 1.0.3. https://CRAN.R-project.org/package=shiny

Winston Chang and Barbara Borges Ribeiro (2017). shinydashboard: Create Dashboards with 'Shiny'. R package version 0.6.1. https://CRAN.R-project.org/package=shinydashboard

Paul Murrell (2009). Importing Vector Graphics: The grImport Package for R. Journal of Statistical Software, 30(4), 1-37. URL http://www.jstatsoft.org/v30/i04/.

Jeroen Ooms (2017). rsvg: Render SVG Images into PDF, PNG, PostScript, or Bitmap Arrays. R package version 1.1. https://CRAN.R-project.org/package=rsvg

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Yihui Xie (2016). DT: A Wrapper of the JavaScript Library 'DataTables'. R package version 0.2. https://CRAN.R-project.org/package=DT

Baptiste Auguie (2016). gridExtra: Miscellaneous Functions for "Grid" Graphics. R package version 2.2.1. https://CRAN.R-project.org/package=gridExtra

Andrie de Vries and Brian D. Ripley (2016). ggdendro: Create Dendrograms and Tree Diagrams Using 'ggplot2'. R package version 0.1-20. https://CRAN.R-project.org/package=ggdendro

Langfelder P and Horvath S, WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics 2008, 9:559 doi:10.1186/1471-2105-9-559

Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/.

Simon Urbanek and Jeffrey Horner (2015). Cairo: R graphics device using cairo graphics library for creating high-quality bitmap (PNG, JPEG, TIFF), vector (PDF, SVG, PostScript) and display (X11 and Win32) output. R package version 1.5-9. https://CRAN.R-project.org/package=Cairo

R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Duncan Temple Lang and the CRAN Team (2017). XML: Tools for Parsing and Generating XML Within R and S-Plus. R package version 3.98-1.9. https://CRAN.R-project.org/package=XML

Carson Sievert, Chris Parmer, Toby Hocking, Scott Chamberlain, Karthik Ram, Marianne Corvellec and Pedro Despouy (NA). plotly: Create Interactive Web Graphics via 'plotly.js'. https://plot.ly/r, https://cpsievert.github.io/plotly_book/, https://github.com/ropensci/plotly.

Matt Dowle and Arun Srinivasan (2017). data.table: Extension of 'data.frame'. R package version 1.10.4. https://CRAN.R-project.org/package=data.table

R. Gentleman, V. Carey, W. Huber and F. Hahne (2017). genefilter: genefilter: methods for filtering genes from high-throughput experiments. R package version 1.58.1.

Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/.

Almende B.V., Benoit Thieurmel and Titouan Robert (2017). visNetwork: Network Visualization using 'vis.js' Library. R package version 2.0.1. https://CRAN.R-project.org/package=visNetwork

Lori Shepherd and Martin Morgan (2020). BiocFileCache: Manage Files Across Sessions. R package version 1.12.1.

See Also

norm_data, aggr_rep, filter_data, spatial_hm, submatrix, adj_mod, matrix_hm, network, return_feature, update_feature, shiny_shm, custom_shiny

Examples

```
#' ## The example data included in this package come from an RNA-seq analysis on
#' ## development of 7 chicken organs under 9 time points (Cardoso-Moreira et al. 2019).
#' ## The complete raw count data are downloaded using the R package ExpressionAtlas
#' ## (Keays 2019) with the accession number "E-MTAB-6769".
#'
#' # Access example count data.
#' count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
#' package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
#' count.chk[1:3, 1:5]
#'
#' # A targets file describing spatial features and variables is made based on the
#' # experiment design.
```

```
#' target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
#' package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
#' # Every column in example data 2 corresponds with a row in the targets file.
#' target.chk[1:5, ]
#' # Store example data in "SummarizedExperiment".
#' library(SummarizedExperiment)
#' se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)
#'
#' # Normalize data.
#' se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)
#'
#' # Aggregate replicates of "spatialFeature_variable", where spatial features are organs
#' # and variables are ages.
#' se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
#' aggr='mean')
#' assay(se.chk.aggr)[1:3, 1:3]
#'
#' # Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
#' # of variance (CV) between 0.2 and 100 are retained.
#' se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
#' pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)
#'
#' # The chicken aSVG downloaded from the EBI aSVG repository (https://github.com/ebi-gene-
#' # expression-group/anatomogram/tree/master/src/svg) is included in this package and
#' # accessed as below.
#' svg.chk <- system.file("extdata/shinyApp/data", "gallus_gallus.svg",
#' package="spatialHeatmap")
#' # Read the chicken aSVG file.
#' svg.chk <- read_svg(svg.path=svg.chk)
#'
#' # Store assay data and aSVG in an "SHM" class.
#' dat.chk <- SPHM(svg=svg.chk, bulk=se.chk.fil)
#' # Plot spatial heatmaps with gene "ENSGALG00000019846".
#' shm(data=dat.chk, ID='ENSGALG00000019846', legend.r=1.9,
#' legend.nrow=5, sub.title.size=7, ncol=3)
#'
#' # Save SHMs as HTML and video files in the "~/test" directory.
#' \donttest{
#' if (!dir.exists('~/test')) dir.create('~/test')
#' shm(data=dat.chk, ID='ENSGALG00000019846', legend.r=1.9,
#' legend.nrow=5, sub.title.size=7, ncol=3, out.dir='~/test')
#' }
#'
```

---

adj_mod *Computing adjacency matrix and identifying network modules*

Description

This function is designed to identify network modules in a data matrix, e.g. returned by submatrix, which builds on the **WGCNA** algorithm (Langfelder and Horvath 2008; Ravasz et al. 2002). Each

module consists of *i.e.* groups of biomolecules showing most similar coexpression profiles.

Usage

```
adj_mod(
  data,
  assay.na = NULL,
  type = "signed",
  power = if (type == "distance") 1 else 6,
  arg.adj = list(),
  TOMType = "unsigned",
  arg.tom = list(),
  method = "complete",
  ds = 0:3,
  minSize = 15,
  arg.cut = list(),
  dir = NULL
)
```

Arguments

`data`	The subsetted data matrix returned by the function `submatrix`.
`assay.na`	Applicable when `data` is 'SummarizedExperiment' or 'SingleCellExperiment', where multiple assays could be stored. The name of target assay to use.
`type`	The network type, one of 'unsigned', 'signed', 'signed hybrid', or 'distance'. Correlations and distances are transformed as follows: for `type="unsigned"`, `adjacency=|cor|^power`; for `type="signed"`, `adjacency=(0.5 * (1+cor) )^power`; for `type="signed hybrid"`, if cor>0 `adjacency=cor^power`, otherwise adjacency=0; and for `type="distance"`, `adjacency=(1-(dist/max(dist))^2)^power`. Refer to **WGCNA** (Langfelder and Horvath 2008) for more details.
`power`	A numeric of soft thresholding power for generating the adjacency matrix. The default is 1 for `type=='distance'` and 6 for other network types.
`arg.adj`	A list of additional arguments passed to `adjacency`, *e.g.* `list(corFnc='cor')`. The default is an empty list `list()`.
`TOMType`	one of `"none"`, `"unsigned"`, `"signed"`, `"signed Nowick"`, `"unsigned 2"`, `"signed 2"` and `"signed Nowick 2"`. If `"none"`, adjacency will be used for clustering. See `TOMsimilarityFromExpr` for details.
`arg.tom`	A list of additional arguments passed to `TOMsimilarity`, *e.g.* `list(verbose=1)`. The default is an empty list `list()`.
`method`	the agglomeration method to be used. This should be (an unambiguous abbreviation of) one of `"ward"`, `"single"`, `"complete"`, `"average"`, `"mcquitty"`, `"median"` or `"centroid"`.
`ds`	One of 0, 1, 2, or 3. The strigency of module identification. The smaller, the less modules with larger sizes.
`minSize`	The expected minimum module size. The default is 15. Refer to **WGCNA** for more details.

arg.cut	A list of additional arguments passed to cutreeHybrid, *e.g.* list(verbose=2). The default is an empty list list().
dir	The directory to save the results, where the adjacency matrix and module assignments are saved as TSV-format files "adj.txt" and "mod.txt" respectively.

Value

A list containing the adjacency matrix and module assignments.

Details

To identify modules, first a similarity matrix including similarities between any pair of biomolecules is computed using distance or correlation-based similarity metrics. Second, the similarity matrix is transformed into an adjacency matrix. Third, the adjacency matrix is used to calculate a topological overlap matrix (TOM) where shared neighborhood information among biomolecules is used to preserve robust connections, while removing spurious connections. Fourth, the distance transformed TOM is used for hierarchical clustering with the "flashClust" package (Langfelder and Horvath 2012). Fifth, network modules are identified with the "dynamicTreeCut" package (Langfelder, Zhang, and Steve Horvath 2016). Its ds (deepSplit) argument can be assigned integer values from 0 to 3, where higher values increase the stringency of the module identification process. Since this is a coexpression analysis, total samples should be at least 5. Otherwise, identified modules are not reliable.

These procedures are wrapped in adj_mod for convenience. The result is a list containing the adjacency matrix and the final module assignments stored in a data.frame. Since the interactive network feature (see network) used in the downstream visualization performs best on smaller modules, only modules obtained with stringent ds settings (here ds=2 and ds=3) are returned.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Langfelder P and Horvath S, WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics 2008, 9:559 doi:10.1186/1471-2105-9-559 Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/ R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Peter Langfelder, Bin Zhang and with contributions from Steve Horvath (2016). dynamicTreeCut: Methods for Detection of Clusters in Hierarchical Clustering Dendrograms. R package version 1.63-1. https://CRAN.R-project.org/package=dynamicTreeCut Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1 Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. “Gene Expression Across Mammalian Organ Development.” Nature 571 (7766): 505–9

Ravasz, E, A L Somera, D A Mongru, Z N Oltvai, and A L Barabási. 2002. "Hierarchical Organization of Modularity in Metabolic Networks." Science 297 (5586): 1551–5. Dragulescu A, Arendt C (2020). _xlsx: Read, Write, Format Excel 2007 and Excel 97/2000/XP/2003 Files_.

Examples

```
## The example data included in this package come from an RNA-seq analysis on
## development of 7 chicken organs under 9 time points (Cardoso-Moreira et al. 2019).
## The complete raw count data are downloaded using the R package ExpressionAtlas
## (Keays 2019) with the accession number "E-MTAB-6769".

# Access example count data.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is made based on the
# experiment design.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)

# Normalize data.
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)

## Subset the data matrix for gene 'ENSGALG00000019846' and 'ENSGALG00000000112'.
se.sub.mat <- submatrix(data=se.chk.fil, ID=c('ENSGALG00000019846',
'ENSGALG00000000112'), p=0.1)

## Hierarchical clustering.
library(dendextend)
# Static matrix heatmap.
mhm.res <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=TRUE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
# Clusters containing "ENSGALG00000019846".
cut_dendro(mhm.res$rowDendrogram, h=15, 'ENSGALG00000019846')
```

```
# Interactive matrix heatmap.
 matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))


# In case the interactive heatmap is not automatically opened, run the following code snippet.
# It saves the heatmap as an HTML file that is assigned to the "file" argument.

mhm <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
htmlwidgets::saveWidget(widget=mhm, file='mhm.html', selfcontained=FALSE)
browseURL('mhm.html')


## Adjacency matrix and module identification
adj.mod <- adj_mod(data=se.sub.mat)

# The adjacency is a measure of co-expression similarity between genes, where larger
# value denotes higher similarity.
adj.mod[['adj']][1:3, 1:3]

# The modules are identified at four sensitivity levels (ds=0, 1, 2, or 3). From 0 to 3,
# more modules are identified but module sizes are smaller. The 4 sets of module
# assignments are returned in a data frame, where column names are sensitivity levels.
# The numbers in each column are module labels, where "0" means genes not assigned to
# any module.
adj.mod[['mod']][1:3, ]

# Static network graph. Nodes are genes and edges are adjacencies between genes.
# The thicker edge denotes higher adjacency (co-expression similarity) while larger node
# indicates higher gene connectivity (sum of a gene's adjacencies with all its direct
# neighbors). The target gene is labeled by "_target".
network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, adj.min=0,
vertex.label.cex=1.5, vertex.cex=4, static=TRUE)

# Interactive network. The target gene ID is appended "_target".
 network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, static=FALSE)
```

---

aggr_rep *Aggregate "Sample__Condition" Replicates in Data Matrix*

Description

This function aggregates "sample__condition" (see `data` argument) replicates by mean or median. The input data is either a `data.frame` or `SummarizedExperiment`.

Usage

```
aggr_rep(data, assay.na = NULL, sam.factor, con.factor = NULL, aggr = "mean")
```

Arguments

data	**Terms** spatial features: cells, tissues, organs, *etc*; variables: experimental variables such as drug dosage, temperature, time points, *etc*; biomolecules: genes, proteins, metabolites, *etc*; spatial heatmap: SHM. **'SummarizedExperiment'** The assays slot stores the data matrix, where rows and columns are biomolecules and spatial featues respectively. Typically, at least two columns of spatial features and variables are stored in the colData slot respectively. When plotting SHMs, only identical spatial features between the data and aSVG will be colored according to the expression values of chosen biomolecules. Replicates of the same type in these two columns should be identical, *e.g.* "tissueA", "tissueA" rather than "tissueA1", "tissueA2". If column names in the assays slot follow the "spatialFeature__variable" scheme, *i.e.* spatial features and variables are concatenated by double underscore, then the colData slot is not required at all. If the data do not have experiment variables, the variable column in colData or the double underscore scheme is not required. **'data.frame'** Rows and columns are biomolecules and spatial featues respectively. If there are experiment variables, the column names should follow the naming scheme "spatialFeature__variable". Otherwise, the column names should only include spatial features. The double underscore is a reserved string for specific purposes in spatialHeatmap, and thus should be avoided for naming spatial feature or variables. A column of biomolecule description can be included. This is only applicable in the interactive network graph (see network), where mousing over a node displays the corresponding description. **vector** In the function shm, the data can be provided in a numeric vector for testing with a single gene. If so, the naming schme of the vector is the same with the data.frame. **Multiple variables** For plotting SHMs, multiple variables contained in the data can be combined into a composite one, and the composite variable will be treated as a regular single variable. See the vignette for more details by running browseVignettes('spatialHeatmap') in R.
assay.na	The name of target assay to use when data is SummarizedExperiment.
sam.factor	The column name corresponding to spatial features in colData of SummarizedExperiment. If the column names in the assay slot already follows the scheme "spatialFeature__variable", then the colData slot is not required and accordingly this argument could be NULL.
con.factor	The column name corresponding to experimental variables in colData of SummarizedExperiment. It can be NULL if column names of in the assay slot already follows the scheme "spatialFeature__variable", or no variable is associated with the data.
aggr	Aggregate "sample__condition" replicates by "mean" or "median". The default is "mean". If the data argument is a SummarizedExperiment, the "sam-

ple__condition" replicates are internally formed by connecting samples and conditions with "__" in `colData` slot, and are subsequently replace the original column names in `assay` slot. If no condition specified to `con.factor`, the data are aggregated by sample replicates. If "none", no aggregation is applied.

Value

The returned value is the same class with the input data, a `data.frame` or `SummarizedExperiment`. In either case, the column names of the data matrix follows the "sample__condition" scheme.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Morgan M, Obenchain V, Hester J, Pagès H (2022). SummarizedExperiment: SummarizedExperiment container. R package version 1.28.0, <https://bioconductor.org/packages/SummarizedExperiment>. R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 McCarthy, Davis J., Chen, Yunshun, Smyth, and Gordon K. 2012. "Differential Expression Analysis of Multifactor RNA-Seq Experiments with Respect to Biological Variation." Nucleic Acids Research 40 (10): 4288–97 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x

Examples

```
## Two example data sets are showcased for the data formats of "data.frame" and
## "SummarizedExperiment" respectively. Both come from an RNA-seq analysis on
## For conveninece, they are included in this package. The complete raw count data are
## downloaded using the R package ExpressionAtlas (Keays 2019) with the accession
## number "E-MTAB-6769".

# Access example data 1.
df.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken_simple.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t', check.names=FALSE)

# Column names follow the naming scheme
# "spatialFeature__variable".
df.chk[1:3, ]

# A column of gene description can be optionally appended.
ann <- paste0('ann', seq_len(nrow(df.chk))); ann[1:3]
```

```
df.chk <- cbind(df.chk, ann=ann)
df.chk[1:3, ]

# Access example data 2.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is required for example
# data 2, which should be made based on the experiment design.

# Access the targets file.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data 2 in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)
# The "rowData" slot can optionally store a data frame of gene annotation.
rowData(se.chk) <- DataFrame(ann=ann)

# Normalize data.
df.chk.nor <- norm_data(data=df.chk, norm.fun='CNF', log2.trans=TRUE)
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
df.chk.aggr <- aggr_rep(data=df.chk.nor, aggr='mean')
df.chk.aggr[1:3, ]

se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
df.chk.fil <- filter_data(data=df.chk.aggr, pOA=c(0.01, 5), CV=c(0.2, 100))
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)
```

`aSVG.remote.repo` *A list of URLs of remote aSVG repos*

Description

A list of URLs of remote aSVG repos, *i.e. EBI anatomogram and spatialHeatmap_aSVG_Repository*.

Usage

```
data(aSVG.remote.repo)
```

Format

A list.

Source

EBI anatomogram spatialHeatmap_aSVG_Repository

References

https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg https://github.com/jianhaizhang/SHM_SVG

Examples

```
data(aSVG.remote.repo)
aSVG.remote.repo
```

---

cell_group *Add manual cell group labels to SingleCellExperiment*

Description

Add manually created cell group labels in a `data.frame` to `SingleCellExperiment`.

Usage

```
cell_group(sce, df.group, cell, cell.group)
```

Arguments

`sce`	A `SingleCellExperiment`.
`df.group`	A `data.frame` of manually created single cell groups. At least two columns are required, corresponding to cell identifiers that are present in `rownames(colData(sce))` and the manually created group labels respectively.
`cell`	The column name in `df.group` indicating cells.
`cell.group`	The column name in `df.group` indicating cell group labels.

Value

An object of `SingleCellExperiment`.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Morgan M, Obenchain V, Hester J, Pagès H (2022). SummarizedExperiment: SummarizedExperiment container. R package version 1.26.1, https://bioconductor.org/packages/SummarizedExperiment

Examples

```
set.seed(10); library(SummarizedExperiment)
# Read single cell data.
sce.pa <- system.file("extdata/shinyApp/data", "cell_mouse_brain.rds", package="spatialHeatmap")
sce <- readRDS(sce.pa)
# Quality control, normalization, dimensionality reduction on the single cell data.
sce.dimred <- process_cell_meta(sce)
# Read manual cell group labels.
manual.clus.mus.sc.pa <- system.file("extdata/shinyApp/data", "manual_cluster_mouse_brain.txt", package="spatial
manual.clus.mus.sc <- read.table(manual.clus.mus.sc.pa, header=TRUE, sep='\t')
# Include manual cell group labels in "SingleCellExperiment".
sce.clus <- cell_group(sce=sce.dimred, df.group=manual.clus.mus.sc, cell='cell', cell.group='cluster')
```

---

cluster_cell *Cluster single cells or combination of single cells and bulk*

Description

Cluster only single cell data or combination of single cell and bulk data. Clusters are created by first building a graph, where nodes are cells and edges represent connections between nearest neighbors, then partitioning the graph. The cluster labels are stored in the `cluster` column of `colData` slot of `SingleCellExperiment`.

Usage

```
cluster_cell(
  sce,
  graph.meth = "knn",
  dimred = "PCA",
  dims = 50,
  assay.type = NULL,
  knn.gr = list(k = 10, directed = FALSE),
  snn.gr = list(k = 10),
  cluster = "wt",
  wt.arg = list(steps = 4),
  fg.arg = list(),
  sl.arg = list(spins = 25),
  le.arg = list(),
  eb.arg = list()
)
```

Arguments

sce	The single cell data or combination of single cell and bulk data at log2 scale after dimensionality reduction in form of SingleCellExperiment.
graph.meth	Method to build a nearest-neighbor graph, snn (see makeSNNGraph) or knn (default, see makeKNNGraph). The clusters are detected by first creating a nearest neighbor graph using snn or knn then partitioning the graph.
dimred	A string of PCA (default) or UMAP specifying which reduced dimensions to use for creating a nearest neighbor graph.
dims	The number of dimensions in 'dimred' to use.
assay.type	If 'dimred=NULL', the assay in 'assayNames(sce)' to use.
knn.gr	Additional arguments in a named list passed to makeKNNGraph.
snn.gr	Additional arguments in a named list passed to makeSNNGraph.
cluster	The clustering method. One of wt (cluster_walktrap, default), fg (cluster_fast_greedy), le (cluster_leading_eigen), sl (cluster_fast_greedy), eb (cluster_edge_betweenness).
wt.arg, fg.arg, sl.arg, le.arg, eb.arg	A named list of arguments passed to wt, fg, le, sl, eb respectively.

Value

A SingleCellExperiment object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Morgan M, Obenchain V, Hester J, Pagès H (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24.0, https://bioconductor.org/packages/SummarizedExperiment. Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. Csardi G, Nepusz T: The igraph software package for complex network research, InterJournal, Complex Systems 1695. 2006. https://igraph.org Lun A (2026). _bluster: Clustering Algorithms for Bioconductor_. doi:10.18129/B9.bioc.bluster <https://doi.org/10.18129/B9.bioc.bluster>. R package version 1.22.0, <https://bioconductor.org/packages/bluster>.

Examples

```
library(SingleCellExperiment)
set.seed(123)  # for reproducibility
# random counts: 1000 genes × 100 cells
counts_mat <- matrix(
  rpois(1000 * 100, lambda = 5), nrow = 1000, ncol = 100,
  dimnames = list(
    paste0("Gene", 1:1000),
```

```
    paste0("Cell", 1:100)
  )
)
sce <- SingleCellExperiment(
  assays = list(counts = counts_mat)
)
sce.qc = qc_cell(sce)
sce.norm <- norm_cell(sce.qc)
sce.dimred <- reduce_dim(sce.norm)

sce.clus <- cluster_cell(sce=sce.dimred, graph.meth='snn', dimred='PCA')
# Clusters.
table(colData(sce.clus)$cluster)


# See details in function "coclus_meta" by running "?coclus_meta".
```

`coclus_opt` *Optimization of co-clustering bulk and single cell data*

Description

This function is specialized in optimizing the co-clustering method that is able to automatically assign bulk tissues to single cells. A vignette is provide at https://jianhaizhang.github.io/spatialHeatmap_supplement/cocluster_optimize.html.

Usage

```
coclus_opt(
  dat.lis,
  df.para,
  df.fil.set,
  batch.par = NULL,
  multi.core.par = NULL,
  wk.dir,
  verbose = TRUE
)
```

Arguments

`dat.lis`	A two-level nested `list`. Each inner `list` consists of three slots of `bulk`, `cell`, and `df.match`, corresponding to bulk data, single cell data, and ground-truth matching between bulk and cells respectively. For example, list(dataset1=list(bulk=bulk.data1, cell=cell.data1, df.match=df.match1), dataset2=list(bulk=bulk.data2, cell=cell.data2, df.match=df.match2)).
`df.para`	A `data.frame` with each row corresponding to a combination of parameter settings in co-clustering.

df.fil.set	A data.frame of filtering settings. E.g. data.frame(p=c(0.1, 0.2), A=rep(1, 2), cv1=c(0.1, 0.2), cv2=rep(50, 2), cutoff=rep(1, 2), p.in.cell=c(0.15, 0.2), p.in.gen=c(0.05, 0.1), row.names=paste0('fil', seq_len(2))).
batch.par	The parameters for first-level parallelization through a cluster scheduler such as SLURM, which is BatchtoolsParam. If NULL (default), the first-level parallelization is skipped.
multi.core.par	The parameters for second-level parallelization, which is MulticoreParam.
wk.dir	The working directory, where results will be saved.
verbose	If TRUE, intermediate messages will be printed.

Value

A data.frame.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Morgan M, Wang J, Obenchain V, Lang M, Thompson R, Turaga N (2022). _BiocParallel: Bioconductor facilities for parallel evaluation_. R package version 1.30.3, <https://github.com/Bioconductor/BiocParallel>.
Li, Song, Masashi Yamada, Xinwei Han, Uwe Ohler, and Philip N Benfey. 2016. "High-Resolution Expression Map of the Arabidopsis Root Reveals Alternative Splicing and lincRNA Regulation." Dev. Cell 39 (4): 508–22 Shahan, Rachel, Che-Wei Hsu, Trevor M Nolan, Benjamin J Cole, Isaiah W Taylor, Anna Hendrika Cornelia Vlot, Philip N Benfey, and Uwe Ohler. 2020. "A Single Cell Arabidopsis Root Atlas Reveals Developmental Trajectories in Wild Type and Cell Identity Mutants." BioRxiv.

Examples

```
# Optimization includes many iterative runs of co-clustering. To reduce runtime, these runs
# are parallelized with the package BiocParallel.
library(BiocParallel)
# To obtain reproducible results, a fixed seed is set for generating random numbers.
set.seed(10)

# Read bulk (S. Li et al. 2016) and two single cell data sets (Shahan et al. 2020), all of
# which are from Arabidopsis root.
blk <- readRDS(system.file("extdata/cocluster/data", "bulk_cocluster.rds",
package="spatialHeatmap")) # Bulk.
sc10 <- readRDS(system.file("extdata/cocluster/data", "sc10_cocluster.rds",
package="spatialHeatmap")) # Single cell.
sc11 <- readRDS(system.file("extdata/cocluster/data", "sc11_cocluster.rds",
package="spatialHeatmap")) # Single cell.
blk; sc10; sc11

# The ground-truth matching between bulk tissue and single cells needs to be defined in form
# of a table so as to classify TRUE/FALSE assignments.
```

```
match.pa <- system.file("extdata/cocluster/data", "true_match_arab_root_cocluster.txt",
package="spatialHeatmap")
df.match.arab <- read.table(match.pa, header=TRUE, row.names=1, sep='\t')
df.match.arab[1:3, ]

# Place the bulk, single cell data, and matching table in a list.
dat.lis <- list(
  dataset1=list(bulk=blk, cell=sc10, df.match=df.match.arab),
  dataset1=list(bulk=blk, cell=sc11, df.match=df.match.arab)
)

# Filtering settings.
df.fil.set <- data.frame(p=c(0.1), A=rep(1, 1), cv1=c(0.1), cv2=rep(50, 1), cutoff=rep(1, 1),
p.in.cell=c(0.15), p.in.gen=c(0.05), row.names=paste0('fil', seq_len(1)))
# Settings in pre-processing include normalization method (norm), filtering (fil). The
# following optimization focuses on settings most relevant to co-clustering, including
# dimension reduction methods (dimred), number of top dimensions for co-clustering (dims),
# graph-building methods (graph), clustering methods (cluster). Explanations of these settings
# are provide in the help file of function "cocluster".
norm <- c('FCT'); fil <- c('fil1'); dimred <- c('UMAP')
dims <- seq(5, 10, 1); graph <- c('knn', 'snn')
cluster <- c('wt', 'fg', 'le')

df.para <- expand.grid(dataset=names(dat.lis), norm=norm, fil=fil, dimred=dimred, dims=dims,
graph=graph, cluster=cluster, stringsAsFactors = FALSE)


# Optimization is performed by calling "coclus_opt", and results to a temporary directory
# "wk.dir".
wk.dir <- normalizePath(tempdir(check=TRUE), winslash="/", mustWork=FALSE)
df.res <- coclus_opt(dat.lis, df.para, df.fil.set, multi.core.par=MulticoreParam(workers=1,
RNGseed=50), wk.dir=wk.dir, verbose=TRUE)
df.res[1:3, ]
```

`cocluster` *Co-clustering bulk and single cell data*

Description

Automatically assigns bulk tissues to single cells through co-clustering.

Shiny App for assigning desired bulk tissues to single cells when tailoring the co-clustering result in co-visualization of bulk and single cell data. The uploaded file is an ".rds" file of a `SingleCellExperiment` object saved by `saveRDS`. See the example below.

Refine the bulk-cell assignments by subsetting the assignments according to a threshold, which is a similarity value between bulk and cells.

Filter single cell data and take overlap genes between cell and bulk data. The bulk data are not filtered as they are only used to obtain overlap genes.

Refine the bulk-cell assignments by including custom bulk-cell assignments.

Usage

```
cocluster(
  bulk,
  cell,
  df.match = NULL,
  min.dim = 11,
  max.dim = 50,
  dimred = "PCA",
  graph.meth = "knn",
  knn.gr = list(),
  snn.gr = list(),
  cluster = "fg",
  wt.arg = list(steps = 4),
  fg.arg = list(),
  sl.arg = list(spins = 25),
  le.arg = list(),
  eb.arg = list(),
  sim.meth = "spearman"
)

desired_bulk_shiny()

filter_asg(res, min.sim = 0)

filter_cell(
  sce,
  bulk = NULL,
  gen.rm = NULL,
  cutoff = 1,
  p.in.cell = 0.4,
  p.in.gen = 0.2,
  com = FALSE,
  verbose = TRUE
)

refine_asg(sce.all, df.desired.bulk = NULL)
```

Arguments

bulk	The bulk data in form of data.frame, SummarizedExperiment, or SingleCellExperiment. They are only used to obtain overlapping genes with single cell data and not filtered. The default is NULL.
cell	The normalized single cell data in form of SingleCellExperiment.
df.match	A data.frame specifying ground-truth matching between cells and bulk, applicable in co-clustering optimization.
min.dim, max.dim	Integer scalars specifying the minimum (min.dim) and maximum (max.dim) number of (principle components) PCs to retain respectively in runPca.se.

dimred — A string of PCA (default) or UMAP specifying which reduced dimensions to use for creating a nearest neighbor graph.

graph.meth — Method to build a nearest-neighbor graph, snn (see makeSNNGraph) or knn (default, see makeKNNGraph). The clusters are detected by first creating a nearest neighbor graph using snn or knn then partitioning the graph.

knn.gr — Additional arguments in a named list passed to makeKNNGraph.

snn.gr — Additional arguments in a named list passed to makeSNNGraph.

cluster — The clustering method. One of wt (cluster_walktrap, default), fg (cluster_fast_greedy), le (cluster_leading_eigen), sl (cluster_fast_greedy), eb (cluster_edge_betweenness).

wt.arg, fg.arg, sl.arg, le.arg, eb.arg — A named list of arguments passed to wt, fg, le, sl, eb respectively.

sim.meth — Method to calculate similarities between bulk and cells in each cocluster when assigning bulk to cells. spearman (default) or pearson.

res — The coclustering results returned by cocluster.

min.sim — The similarity cutoff for filterig bulk-cell assignments, which is a Pearson's or Spearman's correlation coefficient between bulk and cells. Only bulk-cell assignments with similarity values above the thresold would remain. The default is 0.

sce — A SingleCellExperiment of single cell data.

gen.rm — A regular expression of gene identifiers in single cell data to remove before filtering. E.g. mitochondrial, chloroplast and protoplasting-induced genes (^ATCG|^ATCG). The default is NULL.

cutoff — The minmun count of gene expression. The default is 1.

p.in.cell — The proportion cutoff of counts above cutoff in a cell. The default is 0.4.

p.in.gen — The proportion cutoff of counts above cutoff in a gene. The default is 0.2.

com — Logical, if TRUE the returned cell and bulk data are column-wise combined, otherwise they are separated in a list.

verbose — Logical. If TRUE (default), intermediate messages are printed.

sce.all — The coclustering results returned by cocluster.

df.desired.bulk — A "data.frame" of desired bulk for some cells. The cells could be specified by providing x-y axis ranges in an embedding plot ("UMAP", "PCA", "TSNE") returned by plot_dim. E.g. df.desired.bulk <- data.frame(x.min=c(4, -6), x.max=c(5, -5), y.min=c(-2.5, 2), y.max=c(-2, 2.5), desiredSVGBulk=c('CORT', 'STELE'), dimred='UMAP'), where columns x.min, x.max, y.min, y.max, desiredSVGBulk, dimred are required. In this example, cells located in 4 <= x <= 5 and -2.5 <= y <= -2 in the "UMAP" plot are assigned "STELE", and cells located in -6 <= x <= -5 and 2 <= y <= 2.5 in the "UMAP" plot are assigned "CORT".
Alternatively, the "data.frame" could be downloaded from the Shiny app launched by desired_bulk_shiny.
df.desired.bulk is used to tailor the co-clustering results. That is to say additional true bulk-cell assignments are created and included in the final assignments. If these assignments conflict with the co-colustering results the latter would be overwritten.

Value

A list of coclustering results in `SingleCellExperiment` and an `roc` object (relevant in optimization).

A web browser based Shiny app.

A `SingleCellExperiment` of remaining bulk-cell assignments.

A list of filtered single cell data and bulk data, which have common genes.

A `SingleCellExperiment` of remaining bulk-cell assignments.

Details

No argument is required, this function launches the Shiny app directly.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Vacher, Claire-Marie, Helene Lacaille, Jiaqi J. O'Reilly, Jacquelyn Salzbank, Dana Bakalar, Sonia Sebaoui, Philippe Liere, et al. 2021. "Placental Endocrine Function Shapes Cerebellar Development and Social Behavior." Nature Neuroscience 24 (10): 1392–1401. Ortiz, Cantin, Jose Fernandez Navarro, Aleksandra Jurek, Antje Märtin, Joakim Lundeberg, and Konstantinos Meletis. 2020. "Molecular Atlas of the Adult Mouse Brain." Science Advances 6 (26): eabb3446. SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1
R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. Pagès H (2026). _DelayedArray: A unified framework for working transparently with on-disk and in-memory array-like datasets_. doi:10.18129/B9.bioc.DelayedArray <https://doi.org/10.18129/B9.bioc.Delayed
R package version 0.39.1, <https://bioconductor.org/packages/DelayedArray>.

https://www.w3schools.com/graphics/svg_intro.asp

https://shiny.rstudio.com/tutorial/

https://shiny.rstudio.com/articles/datatables.html

https://rstudio.github.io/DT/010-style.html

https://plot.ly/r/heatmaps/

https://www.gimp.org/tutorials/

https://inkscape.org/en/doc/tutorials/advanced/tutorial-advanced.en.html

http://www.microugly.com/inkscape-quickguide/

https://cran.r-project.org/web/packages/visNetwork/vignettes/Introduction-to-visNetwork.html

Winston Chang, Joe Cheng, JJ Allaire, Yihui Xie and Jonathan McPherson (2017). shiny: Web Application Framework for R. R package version 1.0.3. https://CRAN.R-project.org/package=shiny

Winston Chang and Barbara Borges Ribeiro (2017). shinydashboard: Create Dashboards with 'Shiny'. R package version 0.6.1. https://CRAN.R-project.org/package=shinydashboard

Paul Murrell (2009). Importing Vector Graphics: The grImport Package for R. Journal of Statistical Software, 30(4), 1-37. URL http://www.jstatsoft.org/v30/i04/

Jeroen Ooms (2017). rsvg: Render SVG Images into PDF, PNG, PostScript, or Bitmap Arrays. R package version 1.1. https://CRAN.R-project.org/package=rsvg

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Yihui Xie (2016). DT: A Wrapper of the JavaScript Library 'DataTables'. R package version 0.2. https://CRAN.R-project.org/package=DT

Baptiste Auguie (2016). gridExtra: Miscellaneous Functions for "Grid" Graphics. R package version 2.2.1. https://CRAN.R-project.org/package=gridExtra

Andrie de Vries and Brian D. Ripley (2016). ggdendro: Create Dendrograms and Tree Diagrams Using 'ggplot2'. R package version 0.1-20. https://CRAN.R-project.org/package=ggdendro

Langfelder P and Horvath S, WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics 2008, 9:559 doi:10.1186/1471-2105-9-559

Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/

Simon Urbanek and Jeffrey Horner (2015). Cairo: R graphics device using cairo graphics library for creating high-quality bitmap (PNG, JPEG, TIFF), vector (PDF, SVG, PostScript) and display (X11 and Win32) output. R package version 1.5-9. https://CRAN.R-project.org/package=Cairo

R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Duncan Temple Lang and the CRAN Team (2017). XML: Tools for Parsing and Generating XML Within R and S-Plus. R package version 3.98-1.9. https://CRAN.R-project.org/package=XML

Carson Sievert, Chris Parmer, Toby Hocking, Scott Chamberlain, Karthik Ram, Marianne Corvellec and Pedro Despouy (NA). plotly: Create Interactive Web Graphics via 'plotly.js'. https://plot.ly/r, https://cpsievert.github.io/plotly_book/, https://github.com/ropensci/plotly

Matt Dowle and Arun Srinivasan (2017). data.table: Extension of 'data.frame'. R package version 1.10.4. https://CRAN.R-project.org/package=data.table

R. Gentleman, V. Carey, W. Huber and F. Hahne (2017). genefilter: genefilter: methods for filtering genes from high-throughput experiments. R package version 1.58.1.

Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/

Almende B.V., Benoit Thieurmel and Titouan Robert (2017). visNetwork: Network Visualization using 'vis.js' Library. R package version 2.0.1. https://CRAN.R-project.org/package=visNetwork Vacher, Claire-Marie, Helene Lacaille, Jiaqi J. O'Reilly, Jacquelyn Salzbank, Dana Bakalar, Sonia Sebaoui, Philippe Liere, et al. 2021. "Placental Endocrine Function Shapes Cerebellar Development and Social Behavior." Nature Neuroscience 24 (10): 1392–1401. Ortiz, Cantin, Jose Fernandez Navarro, Aleksandra Jurek, Antje Märtin, Joakim Lundeberg, and Konstantinos Meletis. 2020. "Molecular Atlas of the Adult Mouse Brain." Science Advances 6 (26): eabb3446.

Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-

019-0654-x Morgan M, Obenchain V, Hester J, Pagès H (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24.0, https://bioconductor.org/packages/SummarizedExperiment.

Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24.0. https://bioconductor.org/packages/SummarizedExperiment Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." _Nature Methods_, *17*, 137-145. <URL: https://www.nature.com/articles/s41592-019-0654-x> Douglas Bates and Martin Maechler (2021). Matrix: Sparse and Dense Matrix Classes and Methods. R package version 1.4-0. https://CRAN.R-project.org/package=Matrix Vacher CM, Lacaille H, O'Reilly JJ, Salzbank J et al. Placental endocrine function shapes cerebellar development and social behavior. Nat Neurosci 2021 Oct;24(10):1392-1401. PMID: 34400844. Ortiz C, Navarro JF, Jurek A, Märtin A et al. Molecular atlas of the adult mouse brain. Sci Adv 2020 Jun;6(26):eabb3446. PMID: 32637622

Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x Morgan M, Obenchain V, Hester J, Pagès H (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24.0, https://bioconductor.org/packages/SummarizedExperiment.

Examples

```
# To obtain reproducible results, a fixed seed is set for generating random numbers.
set.seed(10); library(SummarizedExperiment)
# Example bulk data of mouse brain for coclustering (Vacher et al 2021).
blk.mus.pa <- system.file("extdata/shinyApp/data", "bulk_mouse_cocluster.rds",
package="spatialHeatmap")
blk.mus <- readRDS(blk.mus.pa)
assay(blk.mus)[1:3, 1:5]

# Example single cell data for coclustering (Ortiz et al 2020).
sc.mus.pa <- system.file("extdata/shinyApp/data", "cell_mouse_cocluster.rds",
package="spatialHeatmap")
sc.mus <- readRDS(sc.mus.pa)
colData(sc.mus)[1:3, , drop=FALSE]


# Normalization: bulk and single cell are combined and normalized, then separated.
mus.lis.nor <- norm_cell(sce=sc.mus, bulk=blk.mus, com=FALSE)

# Aggregate bulk replicates.
blk.mus.aggr <- aggr_rep(data=mus.lis.nor$bulk, assay.na='logcounts', sam.factor='sample',
aggr='mean')
# Filter bulk
blk.mus.fil <- filter_data(data=blk.mus.aggr, pOA=c(0.1, 1), CV=c(0.1, 50), verbose=FALSE)
# Filter cell and subset bulk to genes in cell
blk.sc.mus.fil <- filter_cell(sce=mus.lis.nor$cell, bulk=blk.mus.fil, cutoff=1, p.in.cell=0.1,
p.in.gen=0.01, verbose=FALSE)
# Co-cluster bulk and single cells.
coclus.mus <- cocluster(bulk=blk.sc.mus.fil$bulk, cell=blk.sc.mus.fil$cell, min.dim=12,
dimred='PCA', graph.meth='knn', cluster='wt')
# Co-clustering results. The 'cluster' indicates cluster labels, the 'bulkCell' indicates bulk
```

```
# tissues or single cells, the 'sample' suggests original labels of bulk and cells, the
# 'assignedBulk' refers to bulk tissues assigned to cells with none suggesting un-assigned,
# and the 'similarity' refers to Spearman's correlation coefficients for assignments between
# bulk and cells, which is a measure of assignment strigency.
colData(coclus.mus)

# Filter bulk-cell assignments according a similarity cutoff (min.sim).
coclus.mus <- filter_asg(coclus.mus, min.sim=0.1)

# Tailor bulk-cell assignments in R.
plot_dim(coclus.mus, dim='UMAP', color.by='sample', x.break=seq(-10, 10, 1),
y.break=seq(-10, 10, 1), panel.grid=TRUE)
# Define desired bulk tissues for selected cells.
df.desired.bulk <- data.frame(x.min=c(-8), x.max=c(-3.5), y.min=c(-2.5), y.max=c(0.5),
desiredBulk=c('hippocampus'), dimred='UMAP')
df.desired.bulk
# Tailor bulk-cell assignments.
coclus.mus.tailor <- refine_asg(sce.all=coclus.mus, df.desired.bulk=df.desired.bulk)

# Define desired bulk tissues for selected cells on a Shiny app.
# Save "coclus.mus" using "saveRDS" then upload the saved ".rds" file to the Shiny app.
saveRDS(coclus.mus, file='coclus.mus.rds')

# Start the Shiny app.
desired_bulk_shiny()
```

`com_factor` *Combine Factors in Targets File*

Description

This is a helper function for data/aSVGs involving three or more factors such as sample, time, condition. It combine factors in targets file to make composite factors.

Usage

```
com_factor(se, target, factors2com, sep = ".", factor.new)
```

Arguments

`se`	A `SummarizedExperiment` object.
`target`	A `data.frame` object of targets file.
`factors2com`	A character vector of column names or a numeric vector of column indeces in the targets file. Entries in these columns are combined.
`sep`	The separator in the combined factors. One of _, and . (default).
`factor.new`	The column name of the new combined factors.

Value

If se is provided, a SummarizedExperiment object is returned, where the colData slot contains the new column of combined factors. Otherwise, a data.frame object is returned, where the new column of combined factors is appended.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Attilio, Peter J, Dustin M Snapper, Milan Rusnak, Akira Isaac, Anthony R Soltis, Matthew D Wilkerson, Clifton L Dalgard, and Aviva J Symes. 2021. "Transcriptomic Analysis of Mouse Brain After Traumatic Brain Injury Reveals That the Angiotensin Receptor Blocker Candesartan Acts Through Novel Pathways." Front. Neurosci. 15 (March): 636259

Examples

```
library(SummarizedExperiment)
mus.se.pa <- system.file('extdata/shinyApp/data/mus_brain_vars_se.rds', package='spatialHeatmap')
mus.se <- readRDS(mus.se.pa); targets.info <- colData(mus.se)
targets.new <- com_factor(target=targets.info, factors2com=c('time', 'treatment', 'injury'), factor.new='comDim'
```

---

covis *Co-visualizing spatial heatmaps with single-cell embedding plots*

Description

This function is an extension of shm. It is designed to co-visualize bulk and single-cell data in form of a spatial heatmap (SHM) and embedding plot (PCA, UMAP, TSNE) respectively.

Usage

```
## S4 method for signature 'SPHM'
covis(
  data,
  assay.na = NULL,
  sam.factor = NULL,
  con.factor = NULL,
  ID,
  var.cell = NULL,
  dimred = "PCA",
  cell.group = NULL,
  tar.cell = NULL,
  tar.bulk = NULL,
  size.pt = 1,
```

```
alpha.pt = 0.8,
shape = NULL,
col.idp = FALSE,
decon = FALSE,
profile = TRUE,
charcoal = FALSE,
alpha.overlay = 1,
lay.shm = "gene",
ncol = 2,
h = 0.99,
size.r = 1,
image = NULL,
col.com = c("yellow", "orange", "red"),
col.bar = "selected",
thr = c(NA, NA),
cores = NA,
bar.width = 0.08,
bar.title = NULL,
bar.title.size = 0,
scale = "no",
ft.trans = NULL,
tis.trans = ft.trans,
legend.r = 0,
lgd.plots.size = NULL,
sub.title.size = 11,
sub.title.vjust = 3,
legend.plot = "all",
ft.legend = "identical",
bar.value.size = 10,
legend.plot.title = "Legend",
legend.plot.title.size = 11,
legend.ncol = NULL,
legend.nrow = NULL,
legend.position = "bottom",
legend.direction = NULL,
legend.key.size = 0.02,
legend.text.size = 12,
angle.text.key = NULL,
position.text.key = NULL,
legend.2nd = FALSE,
position.2nd = "bottom",
legend.nrow.2nd = 2,
legend.ncol.2nd = NULL,
legend.key.size.2nd = 0.03,
legend.text.size.2nd = 10,
angle.text.key.2nd = 0,
position.text.key.2nd = "right",
dim.lgd.pos = "bottom",
```

```
  dim.lgd.nrow = 2,
  dim.lgd.key.size = 4,
  dim.lgd.text.size = 13,
  dim.axis.font.size = 8,
  dim.capt.size = 13,
  size.lab.pt = 5,
  hjust.lab.pt = 0.5,
  vjust.lab.pt = 1.5,
  add.feature.2nd = FALSE,
  label = FALSE,
  label.size = 4,
  label.angle = 0,
  hjust = 0,
  vjust = 0,
  opacity = 1,
  key = TRUE,
  line.width = 0.2,
  line.color = "grey70",
  relative.scale = NULL,
  out.dir = NULL,
  animation.scale = 1,
  selfcontained = FALSE,
  aspr = 1,
  video.dim = "640x480",
  res = 500,
  interval = 1,
  framerate = 1,
  bar.width.vdo = 0.1,
  legend.value.vdo = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

`data`	An 'SHM' class that containing the numeric data and aSVG instances for plotting SHMs or co-visualization plots. See SPHM.
`assay.na`	The name of target assay to use when `data` is `SummarizedExperiment`.
`sam.factor`	The column name corresponding to spatial features in `colData` of `SummarizedExperiment`. If the column names in the `assay` slot already follows the scheme "spatialFeature__variable", then the `colData` slot is not required and accordingly this argument could be NULL.
`con.factor`	The column name corresponding to experimental variables in `colData` of `SummarizedExperiment`. It can be NULL if column names of in the `assay` slot already follows the scheme "spatialFeature__variable", or no variable is associated with the data.
`ID`	A character vector of assyed items (*e.g.* genes, proteins) whose abudance values are used to color the aSVG.

var.cell — The column name in the colData slot of SingleCellExperiment that indicates the experimental variables. If NULL, no variables are considered.

dimred — One of PCA, UMAP, TSNE in sce.dimred, specifying which reduced dimension to use in co-visualization plots.

cell.group — Applicable when co-visualizing bulk and single-cell data with cell grouping methods of annodation labels, marker genes, clustering, or manual assignments. A column name in colData(sce.dimred), where one label defines a cell group.

tar.cell — Applicable when co-visualizing bulk and single-cell data with cell-to-bulk mapping. A vector of cell group labels in cell.group which are mapped to tissues through lis.rematch. Cells corresponding to these labels will be colored in the embedding plot while other cells will be grey, and tissues corresponding to these labels will be colored in the SHM while other tissues will be transparent.

tar.bulk — A vector of tissues of interest, which are mapped to single cells through cell group labels with lis.rematch. These tissues will be colored in SHMs while other tissues will be transparent, and cells corresponding to these tissues will be colored embedding plots while other cells will be grey.

size.pt, alpha.pt — The size and alpha value of points in the embedding plot respectively.

shape — A named numeric vector of custom shapes for points in the embedding plot, where the names are the cluster labels and values are from c(0, 2:25, 32:127).

col.idp — Logical, if TRUE, each cell in the embedding plot and spatial feature in the SHM are colored independently according the expression values of chosen biomolecules.

decon — Logical, if TRUE, the cell data will be considered from bulk deconvolution results.

profile — Logical, applicable when co-visualizing bulk and single-cell data. If TRUE, one or multiple biomolecule (e.g. gene) identifiers need to be assigned to ID, and their abundance proifles will be used for coloring in the co-visualization plots. If FALSE, constant colors will be used in the co-visualization plot.

charcoal — Logical, if TRUE the raster image will be turned black and white.

alpha.overlay — The opacity of the raster image under the SHM when superimposing raster images with SHMs. The default is 1.

lay.shm — One of 'gene', 'con', or 'none'. If 'gene', SHMs are organized horizontally by each biomolecule (gene, protein, or metabolite, *etc.*) and variables are sorted under each biomolecule. If 'con', SHMs are organized horizontally by each experiment vairable and biomolecules are sorted under each variable. If 'none', SHMs are organized by the biomolecule order in ID and variables follow the order they appear in data.

ncol — The number of columns to display SHMs, which does not include the legend plot.

h — The height (0-1) of color key and SHM/co-visualization plots in the middle, not including legend plots.

size.r — The scaling ratio (0-1) of tissue section when visualizing the spatially resolved single-cell data.

`image` Name of SpatialImage object to get coordinates for. If NULL (default), will retrieve coordinates for the default image. See `GetTissueCoordinates` in `SeuratObject` for details.

`col.com` A vector of color components used to build the color scale. The default is 'c('yellow', 'orange', 'red')'.

`col.bar` One of 'selected' or 'all', the former uses expression values of `ID` to build the color scale while the latter uses all expression values from the data. The default is 'selected'.

`thr` A two-numeric vector of expression value thresholds (the range of the color bar). The first and the second element will be the minmun and maximum threshold in the color bar respectively. Values above the max or below min will be assigned the same color as the max or min respectively. The default is `c(NA, NA)` and the min and max values in the data will be used. If one needs to change only max or min, the other should be `NA`.

`cores` The number of CPU cores for parallelization. The default is 'NA', and the number of used cores is 1 or 2 depending on the availability.

`bar.width` The width of color bar that ranges from 0 to 1. The default is 0.08.

`bar.title`, `bar.title.size`
The title and title size of the color key.

`scale` One of `no` (default), `selected`, `all`, or `row`, corresponding to no scaling, scaling selected rows as a whole, scaling all rows as a whole, or scaling each row independently.

`ft.trans` A character vector of spatial features that will be set transparent. When features of interest are covered by overlapping features on the top layers and the latter can be set transparent.

`tis.trans` This argument is deprecated and replaced by `ft.trans`.

`legend.r` A numeric (-1 to 1) to adjust the legend plot size.

`lgd.plots.size` A vector of 2 or 3 numeric values that sum up between 0 and 1 (e.g., 'c(0.5, 0.4)'), corresponding to the sizes of legend plots in the co-visualization plot. If there are 2 values, the first and second values correspond to the sizes of the SHM and embedding plots, respectively. If there are 3 values, the first, second, and third values correspond to the sizes of the SHM, single-cell SHM, and embedding plots, respectively.

`sub.title.size` A numeric of the subtitle font size of each individual spatial heatmap. The default is 11.

`sub.title.vjust`
A numeric of vertical adjustment for subtitle. The default is `2`.

`legend.plot` A vector of suffix(es) of aSVG file name(s) such as `c('shm1', 'shm2')`. Only aSVG(s) whose suffix(es) are assigned to this arugment will have a legend plot on the right. The default is `all` and each aSVG will have a legend plot. If NULL, no legend plot is shown.

`ft.legend` One of "identical", "all", or a character vector of tissue/spatial feature identifiers from the aSVG file. The default is "identical" and all the identical/matching tissues/spatial features between the data and aSVG file are colored in the legend plot. If "all", all tissues/spatial features in the aSVG are shown. If a vector, only the tissues/spatial features in the vector are shown.

`bar.value.size` A numeric of value size in the y-axis of the color bar. The default is 10.

`legend.plot.title`
: The title of the legend plot. The default is 'Legend'.

`legend.plot.title.size`
: The title size of the legend plot. The default is 11.

`legend.ncol` An integer of the total columns of keys in the legend plot. The default is NULL. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of shown spatial features.

`legend.nrow` An integer of the total rows of keys in the legend plot. The default is NULL. It is only applicable to the legend plot. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of matching spatial features.

`legend.position`
: the default position of legends ("none", "left", "right", "bottom", "top", "inside")

`legend.direction`
: layout of items in legends ("horizontal" or "vertical")

`legend.key.size`
: A numeric of the legend key size ("npc"), applicable to the legend plot. The default is 0.02.

`legend.text.size`
: A numeric of the legend label size, applicable to the legend plot. The default is 12.

`angle.text.key` Key text angle in legend plots. The default is NULL, equivalent to 0.

`position.text.key`
: The position of key text in legend plots, one of 'top', 'right', 'bottom', 'left'. Default is NULL, equivalent to 'right'.

`legend.2nd` Logical. If 'TRUE', the secondary legend is added to each SHM, which are the numeric values of each colored spatial features. The default its 'FALSE'. Only applies to the static image.

`position.2nd` The position of the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left', or a two-component numeric vector. The default is 'bottom'. Applies to images and videos.

`legend.nrow.2nd`
: An integer of rows of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.ncol.2nd`
: An integer of columns of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.key.size.2nd`
: A numeric of legend key size in SHMs. The default is 0.03. Applies to static images and videos.

`legend.text.size.2nd`
: A numeric of the secondary legend text size in SHMs. The default is 10. Applies to static images and videos.

`angle.text.key.2nd`
: Angle of the key text in the secondary legend in SHMs. Default is 0. Applies to static images and videos.

`position.text.key.2nd`
: The position of key text in the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left'. Default is 'right'. Applies to static images and videos.

`dim.lgd.pos`
: The position of legends in the embedding plot. One of `top`, `right`, `bottom`, and `left`.

`dim.lgd.nrow`
: The number of rows in the embedding plot legends.

`dim.lgd.key.size`, `dim.lgd.text.size`
: The key and font size of the embedding plot legends respectively.

`dim.axis.font.size`
: The size of axis font of the embedding plot.

`dim.capt.size`
: The size of caption text in the dimension reduction plot in co-clustering (see `cocluster`). The default is `13`.

`size.lab.pt`
: The size of point labels, applicable when `decon=TRUE`.

`hjust.lab.pt`, `vjust.lab.pt`
: Numeric values to adjust the horizontal and vertical positions of point labels respectively, applicable when `decon=TRUE`.

`add.feature.2nd`
: Logical. If 'TRUE', feature identifiers are added to the secondary legend. The default is FALSE. Applies to static images of SHMs.

`label`
: Logical. If 'TRUE', the same spatial features between numeric data and aSVG are labeled by their identifiers. The default is 'FALSE'. It is useful when spatial features are labeled by similar colors.

`label.size`
: The size of spatial feature labels in legend plots. The default is 4.

`label.angle`
: The angle of spatial feature labels in legend plots. Default is 0.

`hjust`, `vjust`
: The value to horizontally or vertically adjust positions of spatial feature labels in legend plots respectively. Default of both is 0.

`opacity`
: The transparency of colored spatial features in legend plots. Default is 1. If 0, features are totally transparent.

`key`
: Logical. If 'TRUE' (default), keys are added in legend plots. If `label` is TRUE, the keys could be removed.

`line.width`
: The thickness of each shape outline in the aSVG is maintained in spatial heatmaps, *i.e.* the stroke widths in Inkscape. This argument is the extra thickness added to all outlines. Default is 0.2 in case stroke widths in the aSVG are 0.

`line.color`
: A character of the shape outline color. Default is "grey70".

`relative.scale`
: A numeric to adjust the relative sizes between multiple aSVGs. Applicable only if multiple aSVGs are provided. Default is `NULL` and all aSVGs have the same size.

`out.dir`
: The directory to save SHMs as interactive HTML files and videos. Default is 'NULL', and the HTML files and videos are not saved.

`animation.scale`
: A numeric to scale the SHM size in the HTML files. The default is 1, and the height is 550px and the width is calculated according to the original aspect ratio in the aSVG file.

`selfcontained`
: Whether to save the HTML as a single self-contained file (with external resources base64 encoded) or a file with external resources placed in an adjacent directory.

`aspr`
: The aspect ratio (width to height) in the interative HTML files.

`video.dim`
: A single character of the dimension of video frame in form of 'widthxheight', such as '1920x1080', '1280x800', '320x568', '1280x1024', '1280x720', '320x480', '480x360', '600x600', '800x600', '640x480' (default). The aspect ratio of SHMs are decided by `width` and `height`.

`res`
: Resolution of the video in dpi.

`interval`
: The time interval (seconds) between SHM frames in the video. Default is 1.

`framerate`
: An integer of video framerate in frames per seconds. Default is 1. Larger values make the video smoother.

`bar.width.vdo`
: The color bar width (0-1) in videos.

`legend.value.vdo`
: Logical. If 'TRUE', numeric values of colored spatial features are added to the video legend. The default is NULL.

`verbose`
: Logical. If 'TRUE' (default), intermediate messages will be printed.

`...`
: additional element specifications not part of base ggplot2. In general, these should also be defined in the `element tree` argument. Splicing a list is also supported.

Value

A 'SPHM' class.

Details

See the package vignette (`browseVignettes('spatialHeatmap')`).

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

https://www.gimp.org/tutorials/ https://inkscape.org/en/doc/tutorials/advanced/tutorial-advanced.en.html http://www.microugly.com/inkscape-quickguide/ Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1 H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016. Jeroen Ooms (2018). rsvg: Render SVG Images into PDF, PNG, PostScript, or Bitmap Arrays. R package version 1.3. https://CRAN.R-project.org/package=rsvg R. Gentleman, V. Carey, W. Huber and F. Hahne (2017). genefilter: genefilter: methods for filtering

genes from high-throughput experiments. R package version 1.58.1 Paul Murrell (2009). Importing Vector Graphics: The grImport Package for R. Journal of Statistical Software, 30(4), 1-37. URL http://www.jstatsoft.org/v30/i04/ Baptiste Auguie (2017). gridExtra: Miscellaneous Functions for "Grid" Graphics. R package version 2.3. https://CRAN.R-project.org/package=gridExtra R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. RL https://www.R-project.org/ https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg Yu, G., 2020. ggplotify: Convert Plot to 'grob' or 'ggplot' Object. R package version 0.0.5.URLhttps://CRAN.R-project.org/package=ggplotify30 Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 Guangchuang Yu (2020). ggplotify: Convert Plot to 'grob' or 'ggplot' Object. R package version 0.0.5. https://CRAN.R-project.org/package=ggplotify Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Marques A et al. (2016). Oligodendrocyte heterogeneity in the mouse juvenile and adult central nervous system. Science 352(6291), 1326-1329. Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. Vacher, Claire-Marie, Helene Lacaille, Jiaqi J O'Reilly, Jacquelyn Salzbank, Dana Bakalar, Sonia Sebaoui, Philippe Liere, et al. 2021. "Placental Endocrine Function Shapes Cerebellar Development and Social Behavior." Nat. Neurosci. 24 (10). Nature Publishing Group: 1392–1401

Examples

```
# To always obtain the same results, a fixed seed is set.
set.seed(10); library(SummarizedExperiment); library(ggplot2)
# Read example single cell data of mouse brain (Marques et al. (2016)).
sce.pa <- system.file("extdata/shinyApp/data", "cell_mouse_brain.rds",
package="spatialHeatmap")
# Cell group lables are stored in the "label" column in "colData".
sce <- readRDS(sce.pa); colData(sce)[1:3, 1:3]
# Read example bulk data of mouse brain (Vacher et al. 2021).
blk.mus.pa <- system.file("extdata/shinyApp/data", "bulk_mouse_cocluster.rds",
package="spatialHeatmap"); blk.mus <- readRDS(blk.mus.pa)
assay(blk.mus)[1:3, 1:5]

# Jointly normalize bulk and single cell data.
cache.pa <- '~/.cache/shm' # Cache path.
mus.ann.nor <- read_cache(cache.pa, 'mus.ann.nor')
if (is.null(mus.ann.nor)) {
  mus.lis.nor <- norm_cell(sce=sce, bulk=blk.mus, com=FALSE)
  save_cache(dir=cache.pa, overwrite=TRUE, mus.ann.nor)
}
# Separate normalized bulk and single cell data.
blk.mus.nor <- mus.lis.nor$bulk
cell.mus.nor <- mus.lis.nor$cell; colData(cell.mus.nor) <- colData(sce)
# Reduce dimensions (PCA, UMAP, TSNE) for single cell data.
cell.dim <- reduce_dim(cell.mus.nor)
```

```
# Aggregate tissue replicates in bulk data by mean.
blk.mus.aggr <- aggr_rep(blk.mus.nor, sam.factor='sample', aggr='mean')
assay(blk.mus.aggr)[1:2, ]

# Read the aSVG image into an "SVG" object.
svg.mus.brain.pa <- system.file("extdata/shinyApp/data", "mus_musculus.brain.svg",
package="spatialHeatmap")
svg.mus.brain <- read_svg(svg.mus.brain.pa)
tail(attribute(svg.mus.brain)[[1]])[, 1:4]
# Map tissues to cell groups through a "list", where tissue labels are in name slots and
# the corresponding elements are cell group labels.
lis.match.blk <- list(cerebral.cortex=c('cortex.S1'), hypothalamus=c('corpus.callosum',
'hypothalamus'))
# Store bulk and single-cell data, aSVG, and match list in an "SHM" class.
dat.ann.tocell <- SPHM(svg=svg.mus.brain, bulk=blk.mus.aggr, cell=cell.dim,
match=lis.match.blk)
# Co-visualization is created on expression values of gene "Eif5b". The target cell
# groups are "hypothalamus" and "cortex.S1".
covis(data=dat.ann.tocell, ID=c('Cacnb4'), col.idp=TRUE, dimred='TSNE',
cell.group='label', tar.bulk=names(lis.match.blk), bar.width=0.09, dim.lgd.nrow=2,
dim.lgd.text.size=10, h=0.6, legend.r=0.1, legend.key.size=0.01, legend.text.size=10,
legend.nrow=2)

# Save plots to HTML files and videos in the folder "test".
# covis(data=dat.ann.tocell, ID=c('Cacnb4', 'Apod'), col.idp=TRUE, dimred='UMAP',
# cell.group='label', tar.bulk=names(lis.match.blk), bar.width=0.12, bar.value.size=4,
# dim.lgd.nrow=3, dim.lgd.text.size=7, h=0.5, legend.r=1, animation.scale=0.8, aspr=1.1,
# legend.key.size=0.01, legend.text.size=7, legend.nrow=3, legend.nrow.2nd=3,
# out.dir='test', dim.lgd.key.size=2)
# More examples of co-visualization are provided in the package vignette: https://www.bioconductor.org/packages/re
```

`custom_shiny` *Create Customized spatialHeatmap Shiny Apps*

Description

This function creates customized spatialHeatmap Shiny Apps with user-provided data, aSVG files, and default parameters by using the spatialHeatmap Shiny App as the template.

Usage

```
custom_shiny(
  data = NULL,
  db = NULL,
  lis.par = NULL,
  par.tmp = FALSE,
  dld.sgl = NULL,
  dld.mul = NULL,
```

```
  dld.vars = NULL,
  data.tmp = TRUE,
  ldg = NULL,
  gallery = NULL,
  about = NULL,
  app.dir = "."
)
```

Arguments

data	A nested 'list' of custom data and aSVG files that will be the pre-included examples in the custom App. File paths of each data-aSVG pair should be included in a 'list' that have four slots: 'name', 'display', 'data', and 'svg', e.g. 'lis1 <- list(name='mus.brain', display='Mouse brain (SHM)', data='./mus_brain.txt', svg='./mus_brain.svg')'. The 'name' will be syntactically valide for R code, while the 'display' will be shown on the user interface, so the latter can include special characters. The 'data' and 'svg' is the file path of numeric data and corresponding aSVG respectively. If multiple aSVGs (e.g. growth stages) correspond to a single numeric data set, the respective paths will be stored in a vector in the 'svg' slot (see example below). After store the data-aSVG pairs in separate 'list's, store these 'list's in another 'list' (nested 'list'), e.g. 'list(lis1, lis2)'. The data and aSVGs in the nested 'list' will be copied to the 'data' folder in the custom App.
db	Paired assay data and aSVGs in a tar file that will be used as a backend database. See `write_hdf5`.
lis.par	A 'list' of custom parameters of the Shiny App that will be the default when the custom App is launched. See `par.tmp`. Default is 'NULL' and the default parameters in the spatialHeatmap Shiny App will be inherited.
par.tmp	Logical. If 'TRUE' the default paramters in the spatialHeatmap Shiny App are returned in a 'list', and users can edit these settings then assign the 'list' back to `lis.par`. Note, only the existing values in the 'list' can be edited and the hierarchy of the 'list' should be preserved. Otherwise, it cannot be recognized by the internal program.
dld.sgl	A 'list' of paired data matrix and single aSVG file, which would be downloadable example dataset on the App for testing. The 'list' consists of paths of the data matrix and aSVG file with name slots of 'data' and 'svg' respectively, e.g. `list(data='./data_download.txt', svg='./root_download_shm.svg')`. The specified data and aSVG will copied to the 'data' folder in the App.
dld.mul	A 'list' of paired data matrix and multiple aSVG files, which would be downloadable example dataset on the App for testing. It is the same as 'dld.sgl' except that in the 'svg' slot, multiple aSVG file paths are stored, e.g. 'list(data='./data_download.txt', svg=c('./root_young_download_shm1.svg', './root_old_download_shm2.svg'))'.
dld.vars	A 'list' of paired data matrix and single aSVG file, which would be downloadable data.tmp dataset on the App for testing. It is the same as 'dld.sgl' except that multiple experimental variables are combined in the data matrix (see package viengettes for details.)

`data.tmp` Logical. If 'TRUE' (default), both example data sets in the spatialHeatmap Shiny App and custom data sets in 'data' will be included in the custom App.

`ldg`, `gallery`, `about`
The paths of ".html" files that will be used for the landing, gallery, and about pages in the custom Shiny App respectively. The default is 'NULL', indicating default ".html" files will be used.

`app.dir` The directory to create the Shiny App. Default is current work directory ..

Value

If `par.tmp==TRUE`, the default paramters in spatialHeatmap Shiny App are returned in a 'list'. Otherwise, a customized Shiny App is generated in the path of `app.dir`.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Jeremy Stephens, Kirill Simonov, Yihui Xie, Zhuoer Dong, Hadley Wickham, Jeffrey Horner, reikoch, Will Beasley, Brendan O'Connor and Gregory R. Warnes (2020). yaml: Methods to Convert R Data to YAML and Back. R package version 2.2.1. https://CRAN.R-project.org/package=yaml
Winston Chang, Joe Cheng, JJ Allaire, Yihui Xie and Jonathan McPherson (2017). shiny: Web Application Framework for R. R package version 1.0.3. https://CRAN.R-project.org/package=shiny

Examples

```
# The data sets in spatialHeatmap are used for demonstrations.

## Below are demonstrations of simple usage.
# File paths of one data matrix and one aSVG.
data.path1 <- system.file('extdata/shinyApp/data/expr_mouse.txt', package='spatialHeatmap')
svg.path1 <- system.file('extdata/shinyApp/data/mus_musculus.male.svg',
package='spatialHeatmap')
# Save the file paths in a list with name slots of "name", "display", "data", and "svg".
lis.dat1 <- list(name='Mouse', display='Mouse (SHM)', data=data.path1, svg=svg.path1)
# Create custom Shiny Apps.

if (!dir.exists('~/test_shiny')) dir.create('~/test_shiny')
# Create custom Shiny app by feeding this function these datasets and parameters.
custom_shiny(data=list(lis.dat1), app.dir='~/test_shiny')
# Lauch the app.
shiny::runApp('~/test_shiny/shinyApp')


## Below are demonstrations of advanced usage.

# Paths of one data matrix and two aSVGs (two growth stages).
data.path2 <- system.file('extdata/shinyApp/data/random_data_multiple_aSVGs.txt',
package='spatialHeatmap')
```

```
svg.path2.1 <- system.file('extdata/shinyApp/data/arabidopsis.thaliana_organ_shm1.svg',
package='spatialHeatmap')
svg.path2.2 <- system.file('extdata/shinyApp/data/arabidopsis.thaliana_organ_shm2.svg',
package='spatialHeatmap')
# Save the file paths in a list with name slots of "name", "display", "data", and "svg".
lis.dat2 <- list(name='growthStage', display='Multiple aSVGs (SHM)',data=data.path2,
svg=c(svg.path2.1, svg.path2.2))

# Paths of one data matrix with combined variables and one aSVG.
data.path.vars <- system.file('extdata/shinyApp/data/mus_brain_vars_se_shiny.rds',
package='spatialHeatmap')
svg.path.vars <- system.file('extdata/shinyApp/data/mus_musculus.brain.svg',
package='spatialHeatmap')
# Save the file paths in a list with name slots of "name", "display", "data", and "svg".
lis.dat.vars <- list(name='multiVariables', display='Multiple variables (SHM)',
data=data.path.vars, svg=svg.path.vars)

# Paths of one data matrix and one aSVGs for creating downloadable example data sets.
data.path.dld1 <- system.file('extdata/shinyApp/data/expr_mouse.txt',
package='spatialHeatmap')
svg.path.dld1 <- system.file('extdata/shinyApp/data/mus_musculus.male.svg',
package='spatialHeatmap')
# Save the file paths in a list with name slots of "name", "display", "data", and "svg".
dld.sgl <- list(data=data.path.dld1, svg=svg.path.dld1)

# For demonstration purpose, the same data and aSVGs are used to make the list for creating
# downloadable example dataset of two growth stages.
dld.mul <- list(data=data.path2, svg=c(svg.path2.1, svg.path2.2))

# For demonstration purpose, the same multi-variable data and aSVG are used to create the
# downloadable example multi-variable dataset.
dld.vars <- list(data=data.path.vars, svg=svg.path.vars)

# Retrieve the default parameters in the App template.
lis.par <- custom_shiny(par.tmp=TRUE)
# Change default setting of color scheme in the color key.
lis.par$shm.img['color', ] <- 'yellow,orange,blue'
# The default dataset to show when the app is launched.
lis.par$default.dataset <- 'shoot'

# Store all default data sets in a nested list.
dat.all <- list(lis.dat1, lis.dat2, lis.dat.vars)

# Create custom Shiny Apps.

if (!dir.exists('~/test_shiny')) dir.create('~/test_shiny')
custom_shiny(data=dat.all, lis.par=lis.par, dld.sgl=dld.sgl, dld.mul=dld.mul, dld.vars=dld.vars,
app.dir='~/test_shiny')
# Lauch the App.
shiny::runApp('~/test_shiny/shinyApp')


# The customized Shiny App is able to take database backend as well. Examples are
```

```
# demonstrated in the function "write_hdf5".
```

---

cut_dendro *Cutting dendrograms*

Description

This function is designed to cut dendrograms from heirarchical clustering at a certain height and returns the cluster containing the query biomolecule.

Usage

```
cut_dendro(dendro, h, target)
```

Arguments

dendro	A dendrogram from heirarchical clustering.
h	A numeric of height for cutting the dendrogram.
target	The target biomoleclue.

Value

A vector of the cluster containing the query biomolecule.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Tal Galili (2015). dendextend: an R package for visualizing, adjusting, and comparing trees of hierarchical clustering. Bioinformatics. DOI: 10.1093/bioinformatics/btv428

Examples

```
blk.mus.pa <- system.file("extdata/shinyApp/data", "bulk_mouse_cocluster.rds", package="spatialHeatmap")
blk.mus <- readRDS(blk.mus.pa)
res.hc <- matrix_hm(ID=c('Actr3b'), data=blk.mus, angleCol=60, angleRow=60, cexRow=0.8, cexCol=0.8,
margin=c(10, 6), static=TRUE, arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
cut_dendro(res.hc$rowDendrogram, h=1000, 'Actr3b')
```

cvt_id *Converting gene ids using annotation databases*

Description

This function is designed to convert one type of gene ids to another type, such as Ensembl, Entrez, UniProt.

Usage

```
cvt_id(db, data, from.id, to.id, desc = FALSE, other = NULL)
```

Arguments

db	A character of the annotation database name such as 'org.Hs.eg.db'.
data	A data.frame containing the gene ids in rownames to convert from.
from.id, to.id	The type of ids to convert from ('ENSEMBL', rownames in data) and to ('SYMBOL') (UniProt) respectively.
desc	Logical. If TRUE, the description of each gene will be included.
other	Other information to be added, such as 'c('ENZYME', 'PATH')'. See 'columns(org.Hs.eg.db)'.

Value

A data.frame.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Pagès H, Carlson M, Falcon S, Li N (2022). _AnnotationDbi: Manipulation of SQLite-based annotations in Bioconductor_. R package version 1.60.0, <https://bioconductor.org/packages/AnnotationDbi>.
Morgan M, Obenchain V, Hester J, Pagès H (2022). SummarizedExperiment: SummarizedExperiment container. R package version 1.28. 0, <https://bioconductor.org/packages/SummarizedExperiment>.

Examples

```
library(org.Hs.eg.db)
# Human Ensembl gene ids are rownames in the data frame.
data <- data.frame(tissue1=10:12, tissue2=20:22, row.names=c('ENSG00000006047',
'ENSG00000268433', 'ENSG00000268555'))
data <- cvt_id(db='org.Hs.eg.db', data=data, from.id='ENSEMBL', to.id='SYMBOL', desc=TRUE)
data
```

data_ref *Calculating relative expression values*

Description

This function computes relative expression values for plotting spatial heatmaps (SHMs).

Usage

```
data_ref(se, input.log, output.log = TRUE)
```

Arguments

se	A 'SummarizedExperiment' object containing non-log-transformed data. The 'colData' slot contains a minimum of three columns: 'spFeature' (spatial feature), 'variable' (experiment variable), and 'reference' (reference variable). The first two columns are explained in the 'data' argument of the filter_data function. The 'reference' indicates reference experimental variables that are comma-separated strings such as "variable1,variable2" (see the example below). Relative expression values will be computed based on the references.
input.log	Logical, 'TRUE' or 'FALSE', indicating whether the input assay data are log-transformed or not, repectively. If 'TRUE' this function will not compute relative expression values, since the log-transformed values significantly reduces real relative expression levels. Users are expected to ensure the input assay data are non-log-transformed and then select 'FALSE'.
output.log	Logical. If 'FALSE', the output relative expression values are ratios such as treatment/control, while if 'TRUE', it would be log2 fold changes such as log2(treatment/control).

Value

A 'SummarizedExperiment' object that contains relative expression values.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>

References

Morgan M, Obenchain V, Hester J, Pagès H (2022). SummarizedExperiment: SummarizedExperiment container. R package version 1.28.0, <https://bioconductor.org/packages/SummarizedExperiment>. R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9

Examples

```
library(SummarizedExperiment)
# Access example data.
se <- readRDS(system.file('extdata/shinyApp/data/mouse_organ.rds',
package='spatialHeatmap'))
colData(se)[1:4, 1:3]; assay(se)[1:3, 1:3]
# Data of relative expression values.
se.ref <- data_ref(se, input.log=FALSE)
colData(se.ref)[, 1:3]; assay(se.ref)[1:3, 1:3]
```

`database` *Creat databases for the Shiny App*

Description

The function write_hdf5 is designed to construct the a backend database for the Shiny App (shiny_shm), while read_hdf5 is designed to query the database.

Usage

```
write_hdf5(data, dir = "./data_shm", replace = FALSE)

read_hdf5(file, name = NULL, dir)
```

Arguments

`data`	A nested 'list' of each numeric data and aSVG pair. Each pair consists of four slots: 'name', 'display', 'data', and 'svg', e.g. 'lis1 <- list(name='mus.brain', display='Mouse brain (SHM)', data='./mus_brain.txt', svg='./mus_brain.svg')'. The 'name' contains a syntactically valid entry for R while the 'display' contains an entry for display on the user interface. The 'data' and 'svg' contain paths of numeric data and aSVGs that will be included in the data base respectively. The supported data containers include 'data.frame' and 'SummarizedExperiment'. See the 'data' argument in filter_data for data formats. After formatted, the data should be saved as ".rds" files with the function saveRDS, then assign the corresponding paths the 'data' slot. By contrast, the 'svg' slot contains one or multiple aSVG/raster image file paths. Each numeric data set is first saved in an independent DHF5-based 'SummarizedExperiment' (SE) object, then all these saved SE objects and aSVG/raster images are compressed together into a file "data_shm.tar", i.e. the backend database. In addition, the nested 'list' assigned to 'data' is also included in the database for querying purpose (read_hdf5).
`dir`	The directory for saving assay data and aSVGs in query.
`replace`	If 'TRUE', the existing database with the same name in 'dir' will be replaced.
`file`	The path of a backend database of Shiny App, which is the file of 'data_shm.tar' generated by write_hdf5.
`name`	One or multiple data set names (see the 'data' argument in write_hdf5) in a vector, such as `c('test_leaf', 'test_chicken')`. If 'match', the matching 'list' between assay data and aSVGs will be returned.

Value

'write_hdf5': a file 'data_shm.tar' saved on disk.
'read_hdf5': a nested 'list' of file paths corresponding to data-aSVG pairs.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1
Pagès H, Lawrence M, Aboyoun P (2023). _S4Vectors: Foundation of vector-like and list-like containers in Bioconductor_. doi:10.18129/B9.bioc.S4Vectors <https://doi.org/10.18129/B9.bioc.S4Vectors>, R package version 0.38.1, <https://bioconductor.org/packages/S4Vectors>.
R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/
R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/
Fischer B, Smith M, Pau G (2023). rhdf5: R Interface to HDF5. doi:10.18129/B9.bioc.rhdf5, R package version 2.46.0, https://bioconductor.org/packages/rhdf5
Mustroph, Angelika, M Eugenia Zanetti, Charles J H Jang, Hans E Holtan, Peter P Repetti, David W Galbraith, Thomas Girke, and Julia Bailey-Serres. 2009. "Profiling Translatomes of Discrete Cell Populations Resolves Altered Cellular Priorities During Hypoxia in Arabidopsis." Proc Natl Acad Sci U S A 106 (44): 18843–8
Davis, Sean, and Paul Meltzer. 2007. "GEOquery: A Bridge Between the Gene Expression Omnibus (GEO) and BioConductor." Bioinformatics 14: 1846–7
Gautier, Laurent, Leslie Cope, Benjamin M. Bolstad, and Rafael A. Irizarry. 2004. "Affy—analysis of Affymetrix GeneChip Data at the Probe Level." Bioinformatics 20 (3). Oxford, UK: Oxford University Press: 307–15. doi:10.1093/bioinformatics/btg405
Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas
Huber, W., V. J. Carey, R. Gentleman, S. An ders, M. Carlson, B. S. Carvalho, H. C. Bravo, et al. 2015. "Orchestrating High-Throughput Genomic Analysis Wit H Bioconductor." Nature Methods 12 (2): 115–21. http://www.nature.com/nmeth/journal/v12/n2/full/nmeth.3252.html
Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8
McCarthy, Davis J., Chen, Yunshun, Smyth, and Gordon K. 2012. "Differential Expression Analysis of Multifactor RNA-Seq Experiments with Respect to Biological Variation." Nucleic Acids Research 40 (10): 4288–97
Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9

Examples

```
## Example of a single aSVG instance.

# The example data included in this package come from an RNA-seq analysis on
```

```
# development of 7 chicken organs under 9 time points (Cardoso-Moreira et al. 2019).
# The complete raw count data are downloaded using the R package ExpressionAtlas
# (Keays 2019) with the accession number "E-MTAB-6769".
# Access example count data.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is made based on the
# experiment design.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)

# Indicate spatial features and experiment variables with "spFeature" and "variable"
# in the "colData" slot respectively.
colnames(colData(se.chk))[8] <- 'spFeature'
colnames(colData(se.chk))[6] <- 'variable'
colData(se.chk)[1:2, ]

# Create a temporary directory "db_shm".
dir.db <- file.path(tempdir(check=TRUE), 'db_shm')
if (!dir.exists(dir.db)) dir.create(dir.db)

# Save the assay data in the temporary directory.
chk.dat.pa <- file.path(dir.db, 'test_chicken.rds')
saveRDS(se.chk, file=chk.dat.pa)

# The chicken aSVG downloaded from the EBI aSVG repository (https://github.com/ebi-gene-
# expression-group/anatomogram/tree/master/src/svg) is included in this package and
# accessed as below.
svg.chk <- system.file("extdata/shinyApp/data", "gallus_gallus.svg",
package="spatialHeatmap")
# Store file paths of data and aSVG pair in a list.
dat1 <- list(name='test_chicken', display='Test (chicken SHM)', data=chk.dat.pa, svg=svg.chk)

## Example of multiple aSVG and raster images.

# Two aSVG instances of maize leaves represent two growth stages and each aSVG has a raster
# image counterpart.
# Random numeric data.
pa.leaf <- system.file("extdata/shinyApp/data", 'dat_overlay.txt',
package="spatialHeatmap")
dat.leaf <- read_fr(pa.leaf); dat.leaf[1:2, ]
# Save the assay data in the temporary directory.
leaf.dat.pa <- file.path(dir.db, 'test_leaf.rds')
saveRDS(dat.leaf, file=leaf.dat.pa)

# Paths of the two aSVG files.
```

```
svg.leaf1 <- system.file("extdata/shinyApp/data", 'maize_leaf_shm1.svg',
package="spatialHeatmap")
svg.leaf2 <- system.file("extdata/shinyApp/data", 'maize_leaf_shm2.svg',
package="spatialHeatmap")
# Paths of the two corresponsing raster images.
rst.leaf1 <- system.file("extdata/shinyApp/data", 'maize_leaf_shm1.png',
package="spatialHeatmap")
rst.leaf2 <- system.file("extdata/shinyApp/data", 'maize_leaf_shm2.png',
package="spatialHeatmap")

# Store file paths of data, aSVG, raster images in a list.
dat2 <- list(name='test_leaf', display='Test (maize leaf SHM)', data=leaf.dat.pa,
svg=c(svg.leaf1, svg.leaf2, rst.leaf1, rst.leaf2))

# Store these two data sets in a nested list.
dat.lis <- list(dat1, dat2)


# Save the database in the temporary directory.
write_hdf5(data=dat.lis, dir=dir.db, replace=TRUE)
# Read data-aSVG pair of "test_leaf" from "data_shm.tar".
dat.q <- read_hdf5(file=file.path(dir.db, 'data_shm.tar'), name='test_leaf')
# Numeric data.
dat.leaf <- readRDS(dat.q[[1]]$data)
# aSVG.
svg1 <- dat.q[[1]]$svg[1]; svg2 <- dat.q[[1]]$svg[2]
rst1 <- dat.q[[1]]$svg[3]; rst2 <- dat.q[[1]]$svg[4]
svg.leaf <- read_svg(svg=c(svg1, svg2), raster=c(rst1, rst2))



# Create customized Shiny Apps with the database.
custom_shiny(db=file.path(dir.db, 'data_shm.tar'), app.dir='~/test_shiny')
# Create customized Shiny Apps with the data sets in a nested list.
custom_shiny(data=dat.lis, app.dir='~/test_shiny')
# Run the app.
shiny::runApp('~/test_shiny/shinyApp')
```

`edit_tar` *Edit Targets Files*

Description

Replace existing entries in a chosen column of a targets file with desired ones.

Usage

```
edit_tar(df.tar, column, old, new, sub.row)
```

Arguments

df.tar	The data frame of a targets file.
column	The column to edit, either the column name or an integer of the column index.
old	A vector of existing entries to replace, where the length must be the same with new.
new	A vector of desired entries to replace that in old, where each entry corresponds to a counterpart in old respectively.
sub.row	A vector of integers corresponding to target rows for editing, or a vector of TRUE and FALSE corresponding to each row. Default is all rows in the targets file.

Value

A data.frame.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Mustroph, Angelika, M Eugenia Zanetti, Charles J H Jang, Hans E Holtan, Peter P Repetti, David W Galbraith, Thomas Girke, and Julia Bailey-Serres. 2009. "Profiling Translatomes of Discrete Cell Populations Resolves Altered Cellular Priorities During Hypoxia in Arabidopsis." Proc Natl Acad Sci U S A 106 (44): 18843–8

Examples

```
sh.tar <- system.file('extdata/shinyApp/data/target_arab.txt', package='spatialHeatmap')
target.sh <- read_fr(sh.tar)
target.sh.new <- edit_tar(df.tar=target.sh, column='conditions', old=c('control', 'hypoxia'),
new=c('C', 'H'), sub.row=c(1:12))
```

---

filter_data *Filtering the Data Matrix*

Description

This function is designed to filter the numeric data in form of data.frame or SummarizedExperiment. The filtering builds on two functions pOverA and cv from the package **genefilter** (Gentleman et al. 2018).

Usage

```
filter_data(
  data,
  assay.na = NULL,
  pOA = c(0, 0),
  CV = c(-Inf, Inf),
  top.CV = 1,
  desc = NULL,
  sam.factor = NULL,
  con.factor = NULL,
  file = NULL,
  verbose = TRUE
)
```

Arguments

`data`

Terms spatial features: cells, tissues, organs, *etc*; variables: experimental variables such as drug dosage, temperature, time points, *etc*; biomolecules: genes, proteins, metabolites, *etc*; spatial heatmap: SHM.

'SummarizedExperiment' The `assays` slot stores the data matrix, where rows and columns are biomolecules and spatial featues respectively. Typically, at least two columns of spatial features and variables are stored in the `colData` slot respectively. When plotting SHMs, only identical spatial features between the data and aSVG will be colored according to the expression values of chosen biomolecules. Replicates of the same type in these two columns should be identical, *e.g.* "tissueA", "tissueA" rather than "tissueA1", "tissueA2". If column names in the `assays` slot follow the "spatialFeature__variable" scheme, *i.e.* spatial features and variables are concatenated by double underscore, then the `colData` slot is not required at all. If the data do not have experiment variables, the variable column in `colData` or the double underscore scheme is not required.

'data.frame' Rows and columns are biomolecules and spatial featues respectively. If there are experiment variables, the column names should follow the naming scheme "spatialFeature__variable". Otherwise, the column names should only include spatial features. The double underscore is a reserved string for specific purposes in `spatialHeatmap`, and thus should be avoided for naming spatial feature or variables. A column of biomolecule description can be included. This is only applicable in the interactive network graph (see `network`), where mousing over a node displays the corresponding description.

vector In the function `shm`, the data can be provided in a numeric `vector` for testing with a single gene. If so, the naming schme of the vector is the same with the `data.frame`.

Multiple variables For plotting SHMs, multiple variables contained in the data can be combined into a composite one, and the composite variable will be treated as a regular single variable. See the vignette for more details by running `browseVignettes('spatialHeatmap')` in R.

assay.na	The name of target assay to use when data is SummarizedExperiment.
pOA	Parameters of the filter function pOverA from the package **genefilter** (Gentleman et al. 2018). Genes with expression values >= "A" at the proportion >= "P" of all samples are retained. It is a vector of two numbers, where the first and second is "P" and "A" respectively. The default is c(0, 0), which means no filtering is applied.
CV	Parameters of the filter function cv from the package **genefilter** (Gentleman et al. 2018). Genes with coefficient of variation (CV) between CV1 and CV2 are retained. It is a vector of two numbers, where the first and second is CV1 and CV2 respectively. The default is c(-Inf, Inf), which means no filtering is applied.
top.CV	The proportion (0-1) of top coefficient of variations (CVs). Only genes with CVs in this proportion are kept. *E.g.* if top.CV=0.7, only genes with CVs ranked in the top 70% are retained. Default is 1, which means all genes are retained. Note this argument takes precedence over CV.
desc	A column name in the rowData slot of SummarizedExperiment. The default is NULL. In filter_data, this argument is only applicable if file is specified.
sam.factor	The column name corresponding to spatial features in colData of SummarizedExperiment. If the column names in the assay slot already follows the scheme "spatialFeature__variable", then the colData slot is not required and accordingly this argument could be NULL.
con.factor	The column name corresponding to experimental variables in colData of SummarizedExperiment. It can be NULL if column names of in the assay slot already follows the scheme "spatialFeature__variable", or no variable is associated with the data.
file	The output file name for saving the filtered data matrix in TSV format, which is ready to upload in the Shiny App (see shiny_shm). In this file, the rows are biomoleclues and column names are in the scheme "spatialFeature__variable". If row annotation is provided to desc, it will be appended to the output file. The default is NULL and no file is saved. This argument is applicable only when the data is in SummarizedExperiment and need to be uploaded to the the Shiny App.
verbose	Logical. If TRUE (default), the summary of statistics is printed.

Value

An object of a data.frame or SummarizedExperiment, depending on the input data.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Gentleman, R, V Carey, W Huber, and F Hahne. 2018. "Genefilter: Methods for Filtering Genes from High-Throughput Experiments." http://bioconductor.uib.no/2.7/bioc/html/genefilter.html
Matt Dowle and Arun Srinivasan (2017). data.table: Extension of 'data.frame'. R package version

1.10.4. https://CRAN.R-project.org/package=data.table
Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1
R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x

Examples

```
## Two example data sets are showcased for the data formats of "data.frame" and
## "SummarizedExperiment" respectively. Both come from an RNA-seq analysis on
## For conveninece, they are included in this package. The complete raw count data are
## downloaded using the R package ExpressionAtlas (Keays 2019) with the accession
## number "E-MTAB-6769".

# Access example data 1.
df.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken_simple.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t', check.names=FALSE)

# Column names follow the naming scheme
# "spatialFeature__variable".
df.chk[1:3, ]

# A column of gene description can be optionally appended.
ann <- paste0('ann', seq_len(nrow(df.chk))); ann[1:3]
df.chk <- cbind(df.chk, ann=ann)
df.chk[1:3, ]

# Access example data 2.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is required for example
# data 2, which should be made based on the experiment design.

# Access the targets file.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data 2 in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)
```

```
# The "rowData" slot can optionally store a data frame of gene annotation.
rowData(se.chk) <- DataFrame(ann=ann)

# Normalize data.
df.chk.nor <- norm_data(data=df.chk, norm.fun='CNF', log2.trans=TRUE)
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
df.chk.aggr <- aggr_rep(data=df.chk.nor, aggr='mean')
df.chk.aggr[1:3, ]

se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
df.chk.fil <- filter_data(data=df.chk.aggr, pOA=c(0.01, 5), CV=c(0.2, 100))
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)
```

`matrix_hm` *Hierarchical clustering combined with matrix heatmap*

Description

Given a data matrix returned by `submatrix`, hierarchical clustering is performed on rows and columns respectively and the results are presented in a matrix heatmap, which supports static and interactive modes. In the matrix heatmap, rows and columns are sorted by hierarchical clustering dendrograms and rows of target biomolecules are tagged by black lines. In the interactive heatmap, users can zoom in and out by drawing a rectangle and by double clicking, respectively.

Usage

```
matrix_hm(
  ID,
  data,
  assay.na = NULL,
  scale = "row",
  col = c("yellow", "red"),
  cut.h,
  col.n = 200,
  keysize = 1.8,
  main = NULL,
  title.size = 10,
  cexCol = 1,
  cexRow = 1,
```

```
  angleCol = 45,
  angleRow = 45,
  sep.color = "black",
  sep.width = 0.02,
  static = TRUE,
  margin = c(10, 10),
  arg.lis1 = list(),
  arg.lis2 = list()
)
```

Arguments

ID	A vector of biomolecules of interest in the data matrix.
data	The subsetted data matrix returned by the function submatrix.
assay.na	Applicable when data is 'SummarizedExperiment' or 'SingleCellExperiment', where multiple assays could be stored. The name of target assay to use.
scale	One of 'row', 'column', or 'no' (default), corresponding to scale the heatmap by row, column, or no scaling respectively.
col	A character vector of color ingredients for the color scale. The default is c('yellow', 'orange', 'red').
cut.h	A numeric of the cutting height in the row dendrograms.
col.n	The number of colors in palette.
keysize	A numeric value indicating the size of the color key.
main	The title of the matrix heatmap.
title.size	A numeric value of the title size.
cexCol	A numeric value of column name size. Default is 1.
cexRow	A numeric value of row name size. Default is 1.
angleCol	The angle of column names. The default is 45.
angleRow	The angle of row names. The default is 45.
sep.color	The color of the two lines labeling the row of ID. The default is "black".
sep.width	The width of two lines labeling the row of ID. The default is 0.02.
static	Logical, 'TRUE' and 'FALSE' returns the static and interactive matrix heatmap respectively.
margin	A vector of two numbers, specifying bottom and right margins respectively. The default is c(10, 10).
arg.lis1	A list of additional arguments passed to the heatmap.2 function from "gplots" package. *E.g.* 'list(xlab='sample', ylab='gene')'.
arg.lis2	A list of additional arguments passed to the ggplot function from "ggplot2" package.

Value

A static or interactive matrix heatmap.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1 Andrie de Vries and Brian D. Ripley (2016). ggdendro: Create Dendrograms and Tree Diagrams Using 'ggplot2'. R package version 0.1-20. https://CRAN.R-project.org/package=ggdendro H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016. Carson Sievert (2018) plotly for R. https://plotly-book.cpsievert.me Langfelder P and Horvath S, WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics 2008, 9:559 doi:10.1186/1471-2105-9-559 R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Gregory R. Warnes, Ben Bolker, Lodewijk Bonebakker, Robert Gentleman, Wolfgang Huber Andy Liaw, Thomas Lumley, Martin Maechler, Arni Magnusson, Steffen Moeller, Marc Schwartz and Bill Venables (2019). gplots: Various R Programming Tools for Plotting Data. R package version 3.0.1.1. https://CRAN.R-project.org/package=gplots Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/ Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Matt Dowle and Arun Srinivasan (2019). data.table: Extension of 'data.frame'. R package version 1.12.8. https://CRAN.R-project. org/package=data.table

Examples

```
## The example data included in this package come from an RNA-seq analysis on
## development of 7 chicken organs under 9 time points (Cardoso-Moreira et al. 2019).
## The complete raw count data are downloaded using the R package ExpressionAtlas
## (Keays 2019) with the accession number "E-MTAB-6769".

# Access example count data.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is made based on the
# experiment design.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)
```

```
# Normalize data.
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)

## Subset the data matrix for gene 'ENSGALG00000019846' and 'ENSGALG00000000112'.
se.sub.mat <- submatrix(data=se.chk.fil, ID=c('ENSGALG00000019846',
'ENSGALG00000000112'), p=0.1)

## Hierarchical clustering.
library(dendextend)
# Static matrix heatmap.
mhm.res <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=TRUE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
# Clusters containing "ENSGALG00000019846".
cut_dendro(mhm.res$rowDendrogram, h=15, 'ENSGALG00000019846')

# Interactive matrix heatmap.
 matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))


# In case the interactive heatmap is not automatically opened, run the following code snippet.
# It saves the heatmap as an HTML file that is assigned to the "file" argument.

mhm <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
htmlwidgets::saveWidget(widget=mhm, file='mhm.html', selfcontained=FALSE)
browseURL('mhm.html')


## Adjacency matrix and module identification
adj.mod <- adj_mod(data=se.sub.mat)

# The adjacency is a measure of co-expression similarity between genes, where larger
# value denotes higher similarity.
adj.mod[['adj']][1:3, 1:3]

# The modules are identified at four sensitivity levels (ds=0, 1, 2, or 3). From 0 to 3,
# more modules are identified but module sizes are smaller. The 4 sets of module
# assignments are returned in a data frame, where column names are sensitivity levels.
```

```
# The numbers in each column are module labels, where "0" means genes not assigned to
# any module.
adj.mod[['mod']][1:3, ]

# Static network graph. Nodes are genes and edges are adjacencies between genes.
# The thicker edge denotes higher adjacency (co-expression similarity) while larger node
# indicates higher gene connectivity (sum of a gene's adjacencies with all its direct
# neighbors). The target gene is labeled by "_target".
network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, adj.min=0,
vertex.label.cex=1.5, vertex.cex=4, static=TRUE)

# Interactive network. The target gene ID is appended "_target".
 network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, static=FALSE)
```

`network` *Network graphs*

Description

This function presents a network module returned by `adj_mod` in form of a static or interactive network graph. In the network graph, nodes are biomolecules (genes, proteins, etc) and edges are adjacencies (coexpression similarities) between biomolecules. The thicker edge denotes higher adjacency between nodes while larger node indicates higher connectivity (sum of a node's adjacencies with all its direct neighbours).

In the interactive mode, there is an interactive color scale to denote node connectivity. The color ingredients can only be separated by comma, semicolon, single space, dot, hypen, or underscore. *E.g.* 'yellow,orange,red', which means node connectivity increases from yellow to red. If too many edges (*e.g.*: > 500) are displayed, the app may get crashed, depending on the computer RAM. So the "Adjacency threshold" option sets a threthold to filter out weak edges. Meanwhile, the "Maximun edges" limits the total edges to show. In case a very low adjacency threshold is chosen and introduces too many edges that exceed the "Maximun edges", the App will internally increase the adjacency threshold until the edge total is within the "Maximun edges". The adjacency threshold of 1 leads to no edges. In this case the App wil internally decrease this threshold until the number of edges reaches the "Maximun edges". If adjacency threshold of 0.998 is selected and no edge remains, this App will also internally update the edges to 1 or 2. To maintain acceptable performance, users are advised to choose a stringent threshold (*e.g.* 0.9) initially, then decrease the value gradually. The interactive feature allows users to zoom in and out, or drag a node around. All the node IDs in the network module are listed in "Select by id" in decreasing order according to node connectivity. The ID of a chosen target biomolecule is appended "_target" as a label. By clicking an ID in this list, users can identify the corresponding node in the network. If the input data matrix has annotations for biomolecules, then the annotation can be seen by hovering the cursor over a node.

Usage

```
network(
  ID = NULL,
  mod.lab = NULL,
```

```
  data,
  assay.na = NULL,
  adj.mod,
  ds = "3",
  adj.min = 0,
  con.min = 0,
  desc = NULL,
  node.col = c("turquoise", "violet"),
  edge.col = c("yellow", "blue"),
  vertex.label.cex = 1,
  vertex.cex = 3,
  edge.cex = 3,
  layout = "circle",
  mar.net = c(2, 1, 1, 1),
  mar.key = c(3, 10, 1, 10),
  key.lab.size = 1,
  key.text.size = 1,
  main = NULL,
  static = TRUE,
  dir = NULL,
  ...
)
```

Arguments

ID	A biomolecule of interest. The network module containing this ID will be visualized.
mod.lab	A module label (e.g. 1, 2, 3), the module of which will be visualized.
data	The subsetted data matrix returned by the function submatrix.
assay.na	Applicable when data is 'SummarizedExperiment' or 'SingleCellExperiment', where multiple assays could be stored. The name of target assay to use.
adj.mod	A two-component list returned by adj_mod, consisting of the adjacency matrix and module assignments.
ds	One of '0', '1', '2', or '3' (default), the module identification sensitivity level. The smaller 'ds' indicates larger but less modules while the larger 'ds' denotes smaller but more modules. See function adj_mod for details.
adj.min	A cutoff of adjacency between nodes. Edges with adjacency below this cutoff will not be shown. Default is 0. Applicable to static network.
con.min	A cutoff of node connectivity. Nodes with connectivity below which will not be shown. Default is 0. Applicable to static network.
desc	A column name in the rowData slot of SummarizedExperiment. If provided, when mousing over the nodes in the interactive network, the corresponding description will show up.
node.col	A vector of color ingredients for node color scale in the static image. The default is 'c("turquoise", "violet")', where node connectivity increases from "turquoise" to "violet".

edge.col
: A vector of color ingredients for edge color scale in the static image. The default is ‘c("yellow", "blue")‘, where edge adjacency increases from "yellow" to "blue".

vertex.label.cex
: The size of node label in the static and interactive networks. The default is 1.

vertex.cex
: The size of nodes in the static image. The default is 3.

edge.cex
: The size of edges in the static image. The default is 10.

layout
: The layout of the network in static images, either ‘circle‘ (default) or ‘fr‘. The ‘fr‘ stands for force-directed layout algorithm developed by Fruchterman and Reingold.

mar.net, mar.key
: A four-component numeric vector corresponding to bottom, left, top, and right margins of the network, and color keys respectively.

key.lab.size, key.text.size
: The size of color key label and text respectively.

main
: The title in the static image. Default is NULL.

static
: Logical. ‘TRUE‘ and ‘FALSE‘ return a static and interactive network graphs respectively.

dir
: The directory to save nodes and edges of the module.

...
: Other arguments passed to the generic function plot.default, *e.g.*: asp=1.

Value

A static or interactive network graph.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1 Csardi G, Nepusz T: The igraph software package for complex network research, InterJournal, Complex Systems 1695. 2006. http://igraph.org R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Winston Chang, Joe Cheng, JJ Allaire, Yihui Xie and Jonathan McPherson (2018). shiny: Web Application Framework for R. R package version 1.1.0. https://CRAN.R-project.org/package=shiny Winston Chang and Barbara Borges Ribeiro (2018). shinydashboard: Create Dashboards with 'Shiny'. R package version 0.7.1. https://CRAN.R-project.org/package=shinydashboard Almende B.V., Benoit Thieurmel and Titouan Robert (2018). visNetwork: Network Visualization using 'vis.js' Library. R package version 2.0.4. https://CRAN.R-project.org/package=visNetwork Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan

Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Dragulescu A, Arendt C (2020). _xlsx: Read, Write, Format Excel 2007 and Excel 97/2000/XP/2003 Files_. R package version 0.6.5, <https://CRAN.R-project.org/package=xlsx>.

Examples

```
## The example data included in this package come from an RNA-seq analysis on
## development of 7 chicken organs under 9 time points (Cardoso-Moreira et al. 2019).
## The complete raw count data are downloaded using the R package ExpressionAtlas
## (Keays 2019) with the accession number "E-MTAB-6769".

# Access example count data.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is made based on the
# experiment design.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)

# Normalize data.
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)

## Subset the data matrix for gene 'ENSGALG00000019846' and 'ENSGALG00000000112'.
se.sub.mat <- submatrix(data=se.chk.fil, ID=c('ENSGALG00000019846',
'ENSGALG00000000112'), p=0.1)

## Hierarchical clustering.
library(dendextend)
# Static matrix heatmap.
mhm.res <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=TRUE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
# Clusters containing "ENSGALG00000019846".
```

```
cut_dendro(mhm.res$rowDendrogram, h=15, 'ENSGALG00000019846')

# Interactive matrix heatmap.
 matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))


# In case the interactive heatmap is not automatically opened, run the following code snippet.
# It saves the heatmap as an HTML file that is assigned to the "file" argument.

mhm <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
htmlwidgets::saveWidget(widget=mhm, file='mhm.html', selfcontained=FALSE)
browseURL('mhm.html')


## Adjacency matrix and module identification
adj.mod <- adj_mod(data=se.sub.mat)

# The adjacency is a measure of co-expression similarity between genes, where larger
# value denotes higher similarity.
adj.mod[['adj']][1:3, 1:3]

# The modules are identified at four sensitivity levels (ds=0, 1, 2, or 3). From 0 to 3,
# more modules are identified but module sizes are smaller. The 4 sets of module
# assignments are returned in a data frame, where column names are sensitivity levels.
# The numbers in each column are module labels, where "0" means genes not assigned to
# any module.
adj.mod[['mod']][1:3, ]

# Static network graph. Nodes are genes and edges are adjacencies between genes.
# The thicker edge denotes higher adjacency (co-expression similarity) while larger node
# indicates higher gene connectivity (sum of a gene's adjacencies with all its direct
# neighbors). The target gene is labeled by "_target".
network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, adj.min=0,
vertex.label.cex=1.5, vertex.cex=4, static=TRUE)

# Interactive network. The target gene ID is appended "_target".
 network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, static=FALSE)
```

`norm_cell` *Normalizing single cell data*

Description

A meta function for normalizing single-cell RNA-seq data.

Usage

```
norm_cell(
  sce,
  bulk = NULL,
  cpm = FALSE,
  count.kp = FALSE,
  center.sf = list(block = NULL, mode = c("lowest", "per-block")),
  log.norm = list(assay.type = "counts", log = TRUE, pseudo.count = 1),
  com = FALSE,
  wk.dir = NULL
)
```

Arguments

sce	Single cell count data in form of SingleCellExperiment after quality control, which is returned by qc_cell.
bulk	Bulk tissue count data in form of SingleCellExperiment, SummarizedExperiment, or data.frame.
cpm	Logical. If FALSE (default), the count data are only normalized by normalizeRnaCounts.se. If TRUE, the data are further transformed to counts per million.
count.kp	Logical. If FALSE (default), the count data is discarded and only log2-scale data are kept.
center.sf	Arguments in a named list passed to centerSizeFactors.
log.norm	Arguments in a named list passed to normalizeRnaCounts.se.
com	Logical, if TRUE the returned cell and bulk data are column-wise combined, otherwise they are separated in a list.
wk.dir	The directory path to save normalized data.

Value

A SingleCellExperiment object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. Morgan M, Obenchain V, Hester J, Pagès H (2022). SummarizedExperiment: SummarizedExperiment container. R package version 1.26.1, https://bioconductor.org/packages/SummarizedExperiment
Lun ATL, Kancherla J (2023). "Powering single-cell analyses in the browser with WebAssembly." _Journal of Open Source Software_, *8*(89), 5603. doi:10.21105/joss.05603 <https://doi.org/10.21105/joss.05603>.

Examples

```
library(SingleCellExperiment)
set.seed(123)  # for reproducibility
# random counts: 1000 genes × 100 cells
counts_mat <- matrix(
  rpois(1000 * 100, lambda = 5), nrow = 1000, ncol = 100,
  dimnames = list(
    paste0("Gene", 1:1000),
    paste0("Cell", 1:100)
  )
)
sce <- SingleCellExperiment(
  assays = list(counts = counts_mat)
)
sce.qc = qc_cell(sce)
sce.norm <- norm_cell(sce.qc)
```

---

norm_data *Normalize Sequencing Count Matrix*

Description

This function normalizes sequencing count data in form of `SummarizedExperiment` or `data.frame`. In either class, the columns and rows of the count matix should be samples/conditions and genes respectively.

Usage

```
norm_data(
  data,
  assay.na = NULL,
  norm.fun = "CNF",
  par.list = NULL,
  log2.trans = TRUE,
  data.trans
)
```

Arguments

`data`

Terms spatial features: cells, tissues, organs, *etc*; variables: experimental variables such as drug dosage, temperature, time points, *etc*; biomolecules: genes, proteins, metabolites, *etc*; spatial heatmap: SHM.

'SummarizedExperiment' The `assays` slot stores the data matrix, where rows and columns are biomolecules and spatial featues respectively. Typically, at least two columns of spatial features and variables are stored in the `colData` slot respectively. When plotting SHMs, only identical spatial features between the data and aSVG will be colored according to the expression values of chosen biomolecules. Replicates of the same type in

these two columns should be identical, *e.g.* "tissueA", "tissueA" rather than "tissueA1", "tissueA2". If column names in the `assays` slot follow the "spatialFeature__variable" scheme, *i.e.* spatial features and variables are concatenated by double underscore, then the `colData` slot is not required at all. If the data do not have experiment variables, the variable column in `colData` or the double underscore scheme is not required.

'data.frame' Rows and columns are biomolecules and spatial featues respectively. If there are experiment variables, the column names should follow the naming scheme "spatialFeature__variable". Otherwise, the column names should only include spatial features. The double underscore is a reserved string for specific purposes in `spatialHeatmap`, and thus should be avoided for naming spatial feature or variables. A column of biomolecule description can be included. This is only applicable in the interactive network graph (see `network`), where mousing over a node displays the corresponding description.

vector In the function `shm`, the data can be provided in a numeric `vector` for testing with a single gene. If so, the naming schme of the vector is the same with the `data.frame`.

Multiple variables For plotting SHMs, multiple variables contained in the data can be combined into a composite one, and the composite variable will be treated as a regular single variable. See the vignette for more details by running `browseVignettes('spatialHeatmap')` in R.

`assay.na` The name of target assay to use when `data` is `SummarizedExperiment`.

`norm.fun` Normalizing functions, one of "CNF", "ESF", "VST", "rlog", "none". Specifically, "CNF" stands for `calcNormFactors` from edgeR (McCarthy et al. 2012), and "EST", "VST", and "rlog" is equivalent to `estimateSizeFactors`, `varianceStabilizingTransformation`, and `rlog` from DESeq2 respectively (Love, Huber, and Anders 2014). If "none", no normalization is applied. The default is "CNF" and the output data is processed by `cpm` (Counts Per Million). The parameters of each normalization function are provided through `par.list`.

`par.list` A list of parameters for each normalizing function assigned in `norm.fun`. The default is NULL and `list(method='TMM')`, `list(type='ratio')`, `list(fitType='parametric', blind=TRUE)`, `list(fitType='parametric', blind=TRUE)` is internally set for "CNF", "ESF", "VST", "rlog" respectively. Note the slot name of each element in the `list` is required, *e.g.* `list(method='TMM')` rather than `list('TMM')`.
Complete parameters of "CNF": https://www.rdocumentation.org/packages/edgeR/versions/3.14.0/topics/calcNormFactors
Complete parameters of "ESF": https://www.rdocumentation.org/packages/DESeq2/versions/1.12.3/topics/estimateSizeFactors
Complete parameters of "VST": https://www.rdocumentation.org/packages/DESeq2/versions/1.12.3/topics/varianceStabilizingTransformation
Complete parameters of "rlog": https://www.rdocumentation.org/packages/DESeq2/versions/1.12.3/topics/rlog

`log2.trans` Logical. If `TRUE` (default) and the selected normalization method does not use log2 scale by default ("ESF"), the output data is log2-transformed after normalization. If `FALSE` and the selected normalization method uses log2 scale by

default ("VST", "rlog"), the output data is 2-exponent transformed after normalization.

`data.trans` This argument is deprecated and replaced by `log2.trans`. One of "log2", "exp2", and "none", corresponding to transform the count matrix by "log2", "2-based exponent", and "no transformation" respecitvely. The default is "none".

Value

An object of `SummarizedExperiment` or `data.frame`, depending on the input data.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1
R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ McCarthy, Davis J., Chen, Yunshun, Smyth, and Gordon K. 2012. "Differential Expression Analysis of Multifactor RNA-Seq Experiments with Respect to Biological Variation." Nucleic Acids Research 40 (10): 4288–97 Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 McCarthy, Davis J., Chen, Yunshun, Smyth, and Gordon K. 2012. "Differential Expression Analysis of Multifactor RNA-Seq Experiments with Respect to Biological Variation." Nucleic Acids Research 40 (10): 4288–97 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. “Gene Expression Across Mammalian Organ Development.” Nature 571 (7766): 505–9

See Also

`calcNormFactors` in edgeR, and `estimateSizeFactors`, `varianceStabilizingTransformation`, `rlog` in DESeq2.

Examples

```
## Two example data sets are showcased for the data formats of "data.frame" and
## "SummarizedExperiment" respectively. Both come from an RNA-seq analysis on
## For conveninece, they are included in this package. The complete raw count data are
## downloaded using the R package ExpressionAtlas (Keays 2019) with the accession
## number "E-MTAB-6769".

# Access example data 1.
df.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken_simple.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t', check.names=FALSE)

# Column names follow the naming scheme
# "spatialFeature__variable".
```

```
df.chk[1:3, ]

# A column of gene description can be optionally appended.
ann <- paste0('ann', seq_len(nrow(df.chk))); ann[1:3]
df.chk <- cbind(df.chk, ann=ann)
df.chk[1:3, ]

# Access example data 2.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is required for example
# data 2, which should be made based on the experiment design.

# Access the targets file.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data 2 in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)
# The "rowData" slot can optionally store a data frame of gene annotation.
rowData(se.chk) <- DataFrame(ann=ann)

# Normalize data.
df.chk.nor <- norm_data(data=df.chk, norm.fun='CNF', log2.trans=TRUE)
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
df.chk.aggr <- aggr_rep(data=df.chk.nor, aggr='mean')
df.chk.aggr[1:3, ]

se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
df.chk.fil <- filter_data(data=df.chk.aggr, pOA=c(0.01, 5), CV=c(0.2, 100))
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)
```

`norm_srsc` *Jointly normalization of spatially resolved single cell data and bulk data*

Description

Jointly normalization of spatially resolved single cell data and bulk data

Usage

```
norm_srsc(cell, assay, bulk)
```

Arguments

cell	A Seurat object.
assay	The assay to use for normalization in the spatial single-cell data.
bulk	The bulk assay data.

Value

A list with two elements:

cell The normalized cell data in Seurat object.

bulk The normalized bulk data in SingleCellExperiment.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Hao and Hao et al. Integrated analysis of multimodal single-cell data. Cell (2021) [Seurat V4] Stuart and Butler et al. Comprehensive Integration of Single-Cell Data. Cell (2019) [Seurat V3] Butler et al. Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nat Biotechnol (2018) [Seurat V2] Satija and Farrell et al. Spatial reconstruction of single-cell gene expression data. Nat Biotechnol (2015) [Seurat V1] Pagès H, Lawrence M, Aboyoun P (2022). _S4Vectors: Foundation of vector-like and list-like containers in Bioconductor_. R package version 0.36.1, <https://bioconductor.org/packages/S4Vectors>. Morgan M, Obenchain V, Hester J, Pagès H (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24. 0, https://bioconductor.org/packages/SummarizedExperiment. Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." _Nature Methods_, *17*, 137-145. <https://www.nature.com/articles/s41592-019-0654-x>.

Examples

```
library(Seurat); library(SummarizedExperiment)
# Bulk data.
blk.mus <- readRDS(system.file("extdata/shinyApp/data", "bulk_mouse_cocluster.rds", package="spatialHeatmap"))
assay(blk.mus)[1:3, 1:5]
# Spatial single-cell data.
# library(SeuratData)
```

```
# if (!'stxBrain' %in% InstalledData()$Dataset) InstallData("stxBrain")
# brain <- LoadData("stxBrain", type = "anterior1")
# Joint normalization.
# nor.lis <- norm_srsc(cell=brain, assay='Spatial', bulk=blk.mus)
# Separate bulk and cell data.
# srt.sc <- nor.lis$cell; bulk <- nor.lis$bulk
```

`opt_bar` *Bar plots of co-clustering optimization results.*

Description

Bar plots of co-clustering optimization results.

Usage

```
opt_bar(
  df.res,
  para.na,
  bar.width = 0.8,
  thr = NULL,
  title = NULL,
  title.size = 25,
  xlab = NULL,
  ylab = NULL,
  axis.title.size = 25,
  x.text.size = 25,
  y.text.size = 25,
  x.agl = 80,
  x.vjust = 0.6,
  fill = "#FF6666"
)
```

Arguments

`df.res`	A `data.frame` of co-clustering optimization results.
`para.na`	The targe parameter, which is a column name in `df.res`.
`bar.width`	Width of a single bar.
`thr`	A y-axis threshold, which will be used to draw a horizontal line.
`title`, `title.size`	The plot title and its size.
`xlab`, `ylab`	The x and y axis label respectively.
`axis.title.size`	The size of x and y axis labels.
`x.text.size`, `y.text.size`	The size of x and y axis text.
`x.agl`, `x.vjust`	The angle and vertical position of x-axis text.
`fill`	The color of bars.

Value

An object of ggplot.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Examples

```
set.seed(10)
df.res <- data.frame(dimred=sample(c('PCA', 'UMAP'), 50, replace=TRUE), cluster=sample(c('wt', 'fg', 'le'), 50, re
opt_bar(df.res=df.res, para.na='cluster', ylab='Remaining outcomes')
```

---

opt_setting *Bar plots of co-clustering optimization results.*

Description

Bar plots of co-clustering optimization results.

Usage

```
opt_setting(df.res, nas, summary = "mean")
```

Arguments

df.res	A data.frame of validation results in co-clustering optimization.
nas	The target metrics, which are column names of df.res.
summary	The method to summarize ranks of nas across datasets.

Value

An object of ggplot.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

Examples

```
set.seed(10)
dimred <- c('PCA', 'UMAP'); dims <- seq(5, 80, 15)
graph <- c('knn', 'snn'); cluster <- c('wt', 'fg', 'le')
df.para <- expand.grid(dataset=c('dataset1', 'dataset2'), norm='FCT', fil='fil1', dimred=dimred, dims=dims, graph
df.para$auc <- round(runif(nrow(df.para), 0, 1), 2)
df.para$accuracy <- round(runif(nrow(df.para), 0, 1), 2)
df.para[1:5, ]
opt_setting(df.para, nas=c('auc', 'accuracy'))
```

---

opt_violin *Violin plots of co-clustering validation results*

Description

Violin plots of co-clustering validation results

Usage

```
opt_violin(
  data,
  para.na,
  bar.width = 0.1,
  thr = NULL,
  title = NULL,
  title.size = 25,
  xlab = NULL,
  ylab = NULL,
  axis.title.size = 25,
  x.text.size = 25,
  y.text.size = 25,
  x.agl = 0,
  x.vjust = 0.6
)
```

Arguments

data	A data.frame of co-clustering validation results.
para.na	Targe parameters, which are one or multiple column names in data.
bar.width	Width of the bar.
thr	A y-axis threshold, which will be used to draw a horizontal line.
title, title.size	The plot title and its size.
xlab, ylab	The x and y axis label respectively.
axis.title.size	The size of x and y axis labels.

`x.text.size`, `y.text.size`
: The size of x and y axis text.

`x.agl`, `x.vjust`
: The angle and vertical position of x-axis text.

Value

An object of ggplot.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Examples

```
set.seed(10)
data <- data.frame(auc=round(runif(30, 0, 1), 2), accuracy=round(runif(30, 0, 1), 2))
opt_violin(data=data, para.na=c('auc', 'accuracy'))
```

---

optimal_k

Using the elbow method to find the optimal number of clusters in K-means clustering

---

Description

This function is designed to find the optimal number of clusters in K-means clustering using the elbow method.

Usage

```
optimal_k(data, max.k, title = "Elbow Method for Finding the Optimal k")
```

Arguments

`data`	A numeric `data.frame` scaled with `scale`.
`max.k`	The max number of clusters to consider.
`title`	A character string of the elbow plot title.

Value

A ggplot.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Examples

```
data <- iris[, 1:4]
dat.scl <- scale(data)
optimal_k(dat.scl, 5)
```

---

plot_dim *Embedding plots of single cells/bulk tissues after co-clustering*

Description

Embedding plots of single cells/bulk tissues after co-clustering

Usage

```
plot_dim(
  sce,
  dim = NULL,
  color.by,
  group.sel = NULL,
  row.sel = NULL,
  cocluster.only = TRUE,
  x.break = NULL,
  y.break = NULL,
  panel.grid = FALSE,
  lgd.title.size = 13,
  lgd.key.size = 0.03,
  lgd.text.size = 12,
  point.size = 3,
  bulk.size = 5,
  alpha = 0.7,
  stroke = 0.2,
  bulk.stroke = 1,
  axis.text.size = 10,
  axis.title.size = 11,
  lgd.pos = "right",
  lgd.ncol = 1,
  lgd.l = 0,
  lgd.r = 0.01
)
```

Arguments

sce	A SingleCellExperiment object with reduced dimensions seen by reducedDimNames(sce).
dim	One of PCA, UMAP, TSNE, the method for reducing dimensionality.
color.by	One of the column names in the colData slot of sce.
group.sel	An entry in the color.by column. All cells under this entry are selected as a group to show.
row.sel	A numeric vector of row numbers in the colData slot of sce. The cells corresponding to these rows are highlighted and plotted on top of other cells.
cocluster.only	Logical, only applicable when color.by='cluster'. If TRUE (default), only coclusters (including bulk and cells) are colored and the rest are in gray.
x.break, y.break	Two numeric vectors for x, y axis breaks respectively. E.g. seq(-10, 10, 2). The default is NULL.
panel.grid	Logical. If TRUE, the panel grid will be shown.
lgd.title.size, lgd.key.size, lgd.text.size	The size of legend plot title, legend key, legend text respectively.
point.size, bulk.size	The size of cells and bulk tissues respectively.
alpha	The transparency of cells and bulk tissues. The default is 0.6.
stroke, bulk.stroke	The line width of cells and bulk tissues respectively.
axis.text.size, axis.title.size	The size of axis text and title respectively.
lgd.pos	The legend position, one of top, right, bottom, left.
lgd.ncol	The number of legend columns.
lgd.l, lgd.r	The left and right margins of legends.

Value

An object of ggplot.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016. Morgan M, Obenchain V, Hester J, Pagès H (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24.0, https://bioconductor.org/packages/SummarizedExperiment.

Examples

```
library(SingleCellExperiment)
set.seed(123)  # for reproducibility
# random counts: 1000 genes × 100 cells
counts_mat <- matrix(
  rpois(1000 * 100, lambda = 5), nrow = 1000, ncol = 100,
  dimnames = list(
    paste0("Gene", 1:1000),
    paste0("Cell", 1:100)
  )
)
sce <- SingleCellExperiment(
  assays = list(counts = counts_mat)
)
sce.qc = qc_cell(sce)
sce.norm <- norm_cell(sce.qc)
sce.dimred <- reduce_dim(sce.norm)
plot_dim(sce.dimred, dim='PCA', color.by='detected')
# See function "coclus_meta" by running "?coclus_meta".
```

---

plot_kmeans *Plotting the clusters returned by K-means clustering*

Description

This function is designed to plot the clusters returned by K-means clustering.

Usage

```
plot_kmeans(
  data,
  res,
  query,
  dimred = "TSNE",
  title = "Kmeans Clustering Results"
)
```

Arguments

data	A numeric data.frame scaled with scale.
res	The clustering results returned by kmeans.
query	A rowname in data. The cluster containing this rowname will be labeled.
dimred	The dimension reduction method, one of 'PCA', 'UMAP', or 'TSNE'.
title	The plot title.

Value

A ggplot.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Melville J (2022). _uwot: The Uniform Manifold Approximation and Projection (UMAP) Method for Dimensionality Reduction_. R package version 0.1.14, <https://CRAN.R-project.org/package=uwot>
Jesse H. Krijthe (2015). Rtsne: T-Distributed Stochastic Neighbor Embedding using a Barnes-Hut Implementation, URL: https://github.com/jkrijthe/Rtsne H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Examples

```
set.seed(10)
data <- iris[, 1:4]
rownames(data) <- paste0('gene', seq_len(nrow(data)))
dat.scl <- scale(data)
clus <- kmeans(dat.scl, 6)
plot_kmeans(data=dat.scl, res=clus, query='gene1', dimred='TSNE')
```

---

plot_meta *Meta function for plotting spatial heatmaps or co-visualizing bulk and single cell data*

Description

This is a meta function for plotting spatial heatmaps or co-visualizing bulk and single cell data that takes as input file paths of assay data and aSVG files.

Usage

```
plot_meta(
  svg.path,
  bulk,
  cell = NULL,
  match.lis = NULL,
  assay.na = NULL,
  sam.factor = NULL,
  con.factor = NULL,
  aggr = "mean",
  ID,
  image = NULL,
  var.cell = NULL,
  dimred = "PCA",
  cell.group = NULL,
  tar.cell = NULL,
```

```
tar.bulk = NULL,
size.pt = 1,
alpha.pt = 0.8,
shape = NULL,
col.idp = FALSE,
decon = FALSE,
profile = TRUE,
charcoal = FALSE,
alpha.overlay = 1,
lay.shm = "gene",
ncol = 2,
h = 0.99,
size.r = 1,
col.com = c("yellow", "orange", "red"),
col.bar = "selected",
thr = c(NA, NA),
cores = NA,
bar.width = 0.08,
bar.title = NULL,
bar.title.size = 0,
scale = "no",
ft.trans = NULL,
tis.trans = ft.trans,
legend.r = 0,
lgd.plots.size = NULL,
sub.title.size = 11,
sub.title.vjust = 3,
legend.plot = "all",
ft.legend = "identical",
bar.value.size = 10,
legend.plot.title = "Legend",
legend.plot.title.size = 11,
legend.ncol = NULL,
legend.nrow = NULL,
legend.position = "bottom",
legend.direction = NULL,
legend.key.size = 0.02,
legend.text.size = 12,
angle.text.key = NULL,
position.text.key = NULL,
legend.2nd = FALSE,
position.2nd = "bottom",
legend.nrow.2nd = 2,
legend.ncol.2nd = NULL,
legend.key.size.2nd = 0.03,
legend.text.size.2nd = 10,
angle.text.key.2nd = 0,
position.text.key.2nd = "right",
```

```
  dim.lgd.pos = "bottom",
  dim.lgd.nrow = 2,
  dim.lgd.key.size = 4,
  dim.lgd.text.size = 13,
  dim.axis.font.size = 8,
  dim.capt.size = 13,
  size.lab.pt = 5,
  hjust.lab.pt = 0.5,
  vjust.lab.pt = 1.5,
  add.feature.2nd = FALSE,
  label = FALSE,
  label.size = 4,
  label.angle = 0,
  hjust = 0,
  vjust = 0,
  opacity = 1,
  key = TRUE,
  line.width = 0.2,
  line.color = "grey70",
  relative.scale = NULL,
  out.dir = NULL,
  animation.scale = 1,
  selfcontained = FALSE,
  aspr = 1,
  video.dim = "640x480",
  res = 500,
  interval = 1,
  framerate = 1,
  bar.width.vdo = 0.1,
  legend.value.vdo = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

svg.path	File paths of aSVG files.
bulk, cell	File paths of bulk and single cell data that are saved with saveRDS respectively.
match.lis	The file path of matching list that is saved with saveRDS. See match in SPHM.
assay.na	The name of target assay to use when data is SummarizedExperiment.
sam.factor	The column name corresponding to spatial features in colData of SummarizedExperiment. If the column names in the assay slot already follows the scheme "spatialFeature__variable", then the colData slot is not required and accordingly this argument could be NULL.
con.factor	The column name corresponding to experimental variables in colData of SummarizedExperiment. It can be NULL if column names of in the assay slot already follows the scheme "spatialFeature__variable", or no variable is associated with the data.

aggr — Aggregate "sample__condition" replicates by "mean" or "median". The default is "mean". If the data argument is a SummarizedExperiment, the "sample__condition" replicates are internally formed by connecting samples and conditions with "__" in colData slot, and are subsequently replace the original column names in assay slot. If no condition specified to con.factor, the data are aggregated by sample replicates. If "none", no aggregation is applied.

ID — A character vector of assyed items (*e.g.* genes, proteins) whose abudance values are used to color the aSVG.

image — Name of SpatialImage object to get coordinates for. If NULL (default), will retrieve coordinates for the default image. See GetTissueCoordinates in SeuratObject for details.

var.cell — The column name in the colData slot of SingleCellExperiment that indicates the experimental variables. If NULL, no variables are considered.

dimred — One of PCA, UMAP, TSNE in sce.dimred, specifying which reduced dimension to use in co-visualization plots.

cell.group — Applicable when co-visualizing bulk and single-cell data with cell grouping methods of annodation labels, marker genes, clustering, or manual assignments. A column name in colData(sce.dimred), where one label defines a cell group.

tar.cell — Applicable when co-visualizing bulk and single-cell data with cell-to-bulk mapping. A vector of cell group labels in cell.group which are mapped to tissues through lis.rematch. Cells corresponding to these labels will be colored in the embedding plot while other cells will be grey, and tissues corresponding to these labels will be colored in the SHM while other tissues will be transparent.

tar.bulk — A vector of tissues of interest, which are mapped to single cells through cell group labels with lis.rematch. These tissues will be colored in SHMs while other tissues will be transparent, and cells corresponding to these tissues will be colored embedding plots while other cells will be grey.

size.pt, alpha.pt — The size and alpha value of points in the embedding plot respectively.

shape — A named numeric vector of custom shapes for points in the embedding plot, where the names are the cluster labels and values are from c(0, 2:25, 32:127).

col.idp — Logical, if TRUE, each cell in the embedding plot and spatial feature in the SHM are colored independently according the expression values of chosen biomolecules.

decon — Logical, if TRUE, the cell data will be considered from bulk deconvolution results.

profile — Logical, applicable when co-visualizing bulk and single-cell data. If TRUE, one or multiple biomolecule (e.g. gene) identifiers need to be assigned to ID, and their abundance proifles will be used for coloring in the co-visualization plots. If FALSE, constant colors will be used in the co-visualization plot.

charcoal — Logical, if TRUE the raster image will be turned black and white.

alpha.overlay — The opacity of the raster image under the SHM when superimposing raster images with SHMs. The default is 1.

lay.shm — One of ‘gene‘, ‘con‘, or ‘none‘. If ‘gene‘, SHMs are organized horizontally by each biomolecule (gene, protein, or metabolite, *etc.*) and variables are sorted

under each biomolecule. If 'con', SHMs are organized horizontally by each experiment vairable and biomolecules are sorted under each variable. If 'none', SHMs are organized by the biomolecule order in `ID` and variables follow the order they appear in `data`.

`ncol` The number of columns to display SHMs, which does not include the legend plot.

`h` The height (0-1) of color key and SHM/co-visualization plots in the middle, not including legend plots.

`size.r` The scaling ratio (0-1) of tissue section when visualizing the spatially resolved single-cell data.

`col.com` A vector of color components used to build the color scale. The default is 'c('yellow', 'orange', 'red')'.

`col.bar` One of 'selected' or 'all', the former uses expression values of `ID` to build the color scale while the latter uses all expression values from the data. The default is 'selected'.

`thr` A two-numeric vector of expression value thresholds (the range of the color bar). The first and the second element will be the minmun and maximum threshold in the color bar respectively. Values above the max or below min will be assigned the same color as the max or min respectively. The default is `c(NA, NA)` and the min and max values in the data will be used. If one needs to change only max or min, the other should be `NA`.

`cores` The number of CPU cores for parallelization. The default is 'NA', and the number of used cores is 1 or 2 depending on the availability.

`bar.width` The width of color bar that ranges from 0 to 1. The default is 0.08.

`bar.title`, `bar.title.size`
The title and title size of the color key.

`scale` One of `no` (default), `selected`, `all`, or `row`, corresponding to no scaling, scaling selected rows as a whole, scaling all rows as a whole, or scaling each row independently.

`ft.trans` A character vector of spatial features that will be set transparent. When features of interest are covered by overlapping features on the top layers and the latter can be set transparent.

`tis.trans` This argument is deprecated and replaced by `ft.trans`.

`legend.r` A numeric (-1 to 1) to adjust the legend plot size.

`lgd.plots.size` A vector of 2 or 3 numeric values that sum up between 0 and 1 (e.g., 'c(0.5, 0.4)'), corresponding to the sizes of legend plots in the co-visualization plot. If there are 2 values, the first and second values correspond to the sizes of the SHM and embedding plots, respectively. If there are 3 values, the first, second, and third values correspond to the sizes of the SHM, single-cell SHM, and embedding plots, respectively.

`sub.title.size` A numeric of the subtitle font size of each individual spatial heatmap. The default is 11.

`sub.title.vjust`
A numeric of vertical adjustment for subtitle. The default is `2`.

`legend.plot` A vector of suffix(es) of aSVG file name(s) such as `c('shm1', 'shm2')`. Only aSVG(s) whose suffix(es) are assigned to this arugment will have a legend plot on the right. The default is `all` and each aSVG will have a legend plot. If NULL, no legend plot is shown.

`ft.legend` One of "identical", "all", or a character vector of tissue/spatial feature identifiers from the aSVG file. The default is "identical" and all the identical/matching tissues/spatial features between the data and aSVG file are colored in the legend plot. If "all", all tissues/spatial features in the aSVG are shown. If a vector, only the tissues/spatial features in the vector are shown.

`bar.value.size` A numeric of value size in the y-axis of the color bar. The default is 10.

`legend.plot.title`
The title of the legend plot. The default is 'Legend'.

`legend.plot.title.size`
The title size of the legend plot. The default is 11.

`legend.ncol` An integer of the total columns of keys in the legend plot. The default is NULL. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of shown spatial features.

`legend.nrow` An integer of the total rows of keys in the legend plot. The default is NULL. It is only applicable to the legend plot. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of matching spatial features.

`legend.position`
the default position of legends ("none", "left", "right", "bottom", "top", "inside")

`legend.direction`
layout of items in legends ("horizontal" or "vertical")

`legend.key.size`
A numeric of the legend key size ("npc"), applicable to the legend plot. The default is 0.02.

`legend.text.size`
A numeric of the legend label size, applicable to the legend plot. The default is 12.

`angle.text.key` Key text angle in legend plots. The default is NULL, equivalent to 0.

`position.text.key`
The position of key text in legend plots, one of 'top', 'right', 'bottom', 'left'. Default is NULL, equivalent to 'right'.

`legend.2nd` Logical. If 'TRUE', the secondary legend is added to each SHM, which are the numeric values of each colored spatial features. The default its 'FALSE'. Only applies to the static image.

`position.2nd` The position of the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left', or a two-component numeric vector. The default is 'bottom'. Applies to images and videos.

`legend.nrow.2nd`
An integer of rows of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.ncol.2nd`
: An integer of columns of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.key.size.2nd`
: A numeric of legend key size in SHMs. The default is 0.03. Applies to static images and videos.

`legend.text.size.2nd`
: A numeric of the secondary legend text size in SHMs. The default is 10. Applies to static images and videos.

`angle.text.key.2nd`
: Angle of the key text in the secondary legend in SHMs. Default is 0. Applies to static images and videos.

`position.text.key.2nd`
: The position of key text in the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left'. Default is 'right'. Applies to static images and videos.

`dim.lgd.pos`
: The position of legends in the embedding plot. One of `top`, `right`, `bottom`, and `left`.

`dim.lgd.nrow`
: The number of rows in the embedding plot legends.

`dim.lgd.key.size`, `dim.lgd.text.size`
: The key and font size of the embedding plot legends respectively.

`dim.axis.font.size`
: The size of axis font of the embedding plot.

`dim.capt.size`
: The size of caption text in the dimension reduction plot in co-clustering (see `cocluster`). The default is `13`.

`size.lab.pt`
: The size of point labels, applicable when `decon=TRUE`.

`hjust.lab.pt`, `vjust.lab.pt`
: Numeric values to adjust the horizontal and vertical positions of point labels respectively, applicable when `decon=TRUE`.

`add.feature.2nd`
: Logical. If 'TRUE', feature identifiers are added to the secondary legend. The default is FALSE. Applies to static images of SHMs.

`label`
: Logical. If 'TRUE', the same spatial features between numeric data and aSVG are labeled by their identifiers. The default is 'FALSE'. It is useful when spatial features are labeled by similar colors.

`label.size`
: The size of spatial feature labels in legend plots. The default is 4.

`label.angle`
: The angle of spatial feature labels in legend plots. Default is 0.

`hjust`, `vjust`
: The value to horizontally or vertically adjust positions of spatial feature labels in legend plots respectively. Default of both is 0.

`opacity`
: The transparency of colored spatial features in legend plots. Default is 1. If 0, features are totally transparent.

`key`
: Logical. If 'TRUE' (default), keys are added in legend plots. If `label` is TRUE, the keys could be removed.

`line.width`
: The thickness of each shape outline in the aSVG is maintained in spatial heatmaps, *i.e.* the stroke widths in Inkscape. This argument is the extra thickness added to all outlines. Default is 0.2 in case stroke widths in the aSVG are 0.

line.color	A character of the shape outline color. Default is "grey70".
relative.scale	A numeric to adjust the relative sizes between multiple aSVGs. Applicable only if multiple aSVGs are provided. Default is NULL and all aSVGs have the same size.
out.dir	The directory to save SHMs as interactive HTML files and videos. Default is 'NULL', and the HTML files and videos are not saved.
animation.scale	A numeric to scale the SHM size in the HTML files. The default is 1, and the height is 550px and the width is calculated according to the original aspect ratio in the aSVG file.
selfcontained	Whether to save the HTML as a single self-contained file (with external resources base64 encoded) or a file with external resources placed in an adjacent directory.
aspr	The aspect ratio (width to height) in the interative HTML files.
video.dim	A single character of the dimension of video frame in form of 'widthxheight', such as '1920x1080', '1280x800', '320x568', '1280x1024', '1280x720', '320x480', '480x360', '600x600', '800x600', '640x480' (default). The aspect ratio of SHMs are decided by width and height.
res	Resolution of the video in dpi.
interval	The time interval (seconds) between SHM frames in the video. Default is 1.
framerate	An integer of video framerate in frames per seconds. Default is 1. Larger values make the video smoother.
bar.width.vdo	The color bar width (0-1) in videos.
legend.value.vdo	Logical. If 'TRUE', numeric values of colored spatial features are added to the video legend. The default is NULL.
verbose	Logical. If 'TRUE' (default), intermediate messages will be printed.
...	additional element specifications not part of base ggplot2. In general, these should also be defined in the element tree argument. Splicing a list is also supported.

Value

A 'SPHM' class.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

Examples

```
# Path of mouse brain aSVG.
svg.mus.brain.pa <- system.file("extdata/shinyApp/data", "mus_musculus.brain.svg", package="spatialHeatmap")
# Bulk data path.
blk.mus.pa <- system.file("extdata/shinyApp/data", "bulk_mouse_cocluster.rds", package="spatialHeatmap")
```

```
# Plotting spatial heatmaps.
plot_meta(svg.path=svg.mus.brain.pa, bulk=blk.mus.pa, sam.factor='tissue', aggr='mean', ID=c('AI593442', 'Adora1
```

---

process_cell_meta *Processing single cell RNA-seq count data*

Description

A meta function for processing single cell RNA-seq count data, including quality control, normalization, dimensionality reduction.

Usage

```
process_cell_meta(
  sce,
 qc.metric = list(assay.type = "counts", subsets = list(), altexp.proportions = NULL,
    block = NULL),
  center.sf = list(block = NULL, mode = c("lowest", "per-block")),
  log.norm = list(assay.type = "counts", log = TRUE, pseudo.count = 1),
  choose.hvg = list(assay.type = "logcounts", block = NULL, more.var.args =
   list(transform = TRUE), top = 4000, more.choose.args = list(keep.ties = TRUE)),
  min.dim = 5,
  max.dim = 25,
  pca.arg = list(assay.type = "logcounts", number = 25, block = NULL),
  umap.arg = list(num.dim = 2, reddim.type = "PCA"),
  tsne.arg = list(perplexity = 30, reddim.type = "PCA", output.name = "TSNE"),
  pca = FALSE
)
```

Arguments

sce	Single cell RNA-seq count data in SingleCellExperiment.
qc.metric	Quality control arguments in a named list passed to quickRnaQc.se. Eg: 'list(assay.type = "counts", subsets=list(mito=grepl("^mt", rownames(sce))), altexp.proportions="ERCC")'.
center.sf	Arguments in a named list passed to centerSizeFactors.
log.norm	Arguments in a named list passed to normalizeRnaCounts.se.
choose.hvg	Arguments in a named list passed to chooseRnaHvgs.se.
min.dim, max.dim	Integer scalars specifying the minimum (min.dim) and maximum (max.dim) number of (principle components) PCs to retain respectively in runPca.se.
pca.arg	Arguments in a named list passed to runPca.se.
umap.arg	Arguments in a named list passed to runUmap.se.
tsne.arg	Arguments in a named list passed to runTsne.se.
pca	Logical, if TRUE only the data with reduced dimentionality of PCA is returned. The default is FALSE, and UMAP and TSNE are also returned.

Details

In the QC, compute per-cell QC metrics from an initialized matrix of RNA counts, and use the metrics to suggest filter thresholds to retain high-quality cells. Refer to `filterRnaQcMetrics` in the 'scrapper' package for more details. In the normalization, compute log-normalized expression values after performing scaling normalization of an RNA count matrix. See more details in `normalizeRnaCounts.se` from the scrapper package. In dimensionality reduction, the high-dimensional gene expression data are embedded into a low dimensional space using PCA, tSNE and UMAP. All three embedding result sets are stored in a `SingleCellExperiment` object. Details are seen in `runPca.se`, `runUmap.se`, and `runTsne.se` from 'scrapper'.

Value

A `SingleCellExperiment` object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. McCarthy DJ, Campbell KR, Lun ATL, Willis QF (2017). "Scater: pre-processing, quality control, normalisation and visualisation of single-cell RNA-seq data in R." Bioinformatics, 33, 1179-1186. doi: 10.1093/bioinformatics/btw777. Lun ATL, McCarthy DJ, Marioni JC (2016). "A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor." F1000Res., 5, 2122. doi: 10.12688/f1000research.9501.2.

Examples

```
library(SingleCellExperiment)
set.seed(123)  # for reproducibility
# random counts: 1000 genes × 100 cells
counts_mat <- matrix(
  rpois(1000 * 100, lambda = 5), nrow = 1000, ncol = 100,
  dimnames = list(
    paste0("Gene", 1:1000),
    paste0("Cell", 1:100)
  )
)
sce <- SingleCellExperiment(
  assays = list(counts = counts_mat)
)
sce.dimred <- process_cell_meta(sce)
```

qc_cell *Quality control in single cell data*

Description

A meta function for quality control in single-cell RNA-seq data.

Usage

```
qc_cell(
  sce,
 qc.metric = list(assay.type = "counts", subsets = list(), altexp.proportions = NULL,
    block = NULL)
)
```

Arguments

sce	Raw single cell count data in form of SingleCellExperiment.
qc.metric	Quality control arguments in a named list passed to quickRnaQc.se. Eg: 'list(assay.type = "counts", subsets=list(mito=grepl("^mt", rownames(sce))), altexp.proportions="ERCC")'.

Value

A SingleCellExperiment object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. McCarthy DJ, Campbell KR, Lun ATL, Willis QF (2017). "Scater: pre-processing, quality control, normalisation and visualisation of single-cell RNA-seq data in R." Bioinformatics, 33, 1179-1186. doi: 10.1093/bioinformatics/btw777. Lun ATL, Kancherla J (2023). "Powering single-cell analyses in the browser with WebAssembly." _Journal of Open Source Software_, *8*(89), 5603. doi:10.21105/joss.05603 <https://doi.org/10.21105/joss.05603>.

Examples

```
library(SingleCellExperiment)
set.seed(123)  # for reproducibility
# random counts: 1000 genes × 100 cells
counts_mat <- matrix(
  rpois(1000 * 100, lambda = 5), nrow = 1000, ncol = 100,
```

```
  dimnames = list(
    paste0("Gene", 1:1000),
    paste0("Cell", 1:100)
  )
)
sce <- SingleCellExperiment(
  assays = list(counts = counts_mat)
)
sce.qc = qc_cell(sce)
```

`read_cache` *Read R Objects from Cache*

Description

Read R Objects from Cache

Usage

```
read_cache(dir, name, day = 180, info = FALSE)
```

Arguments

`dir`	The directory path where cached data are located. It should be the path returned by save_cache.
`name`	The name of the object to retrieve, which is one of the entries in the "rname" column returned by setting `info=TRUE`.
`day`	An integer representing cache age in days. If the cache is older than this age, NULL is returned.
`info`	Logical, TRUE or FALSE. If TRUE (default), the information of all tracked files in cache is returned in a table.

Value

An R object retrieved from the cache.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Lori Shepherd and Martin Morgan (2020). BiocFileCache: Manage Files Across Sessions. R package version 1.12.1.

Examples

```
# Save the object "iris" in the default cache "~/.cache/shm".
cache.pa <- save_cache(dir=NULL, overwrite=TRUE, iris)
# Retrieve "iris".
iris1 <- read_cache(cache.pa, 'iris')
```

`read_fr` *Import Data from Tabular Files*

Description

This function reads data from a tabular file, which is a wrapper of fread. If the tabular file contains both character and numeric columns, it is able to maintain the character or numeric attribute for each column in the returned data frame. In addition, it is able to detect separators automatically.

Usage

```
read_fr(input, header = TRUE, sep = "auto", fill = TRUE, check.names = FALSE)
```

Arguments

`input`	The file path.
`header`	One of TRUE, FALSE, or "auto". Default is TRUE. Does the first data line contain column names, according to whether every non-empty field on the first data line is type character? If "auto" or TRUE is supplied, any empty column names are given a default name.
`sep`	The separator between columns. Defaults to the character in the set `[,\t |;:]` that separates the sample of rows into the most number of lines with the same number of fields. Use `NULL` or `""` to specify no separator; i.e. each line a single character column like `base::readLines` does.
`fill`	Logical (default is TRUE). If TRUE then in case the rows have unequal length, blank fields are implicitly filled.
`check.names`	default is `FALSE`. If `TRUE` then the names of the variables in the `data.table` are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by `make.names`) so that they are, and also to ensure that there are no duplicates.

Value

A data frame.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Matt Dowle and Arun Srinivasan (2019). data.table: Extension of 'data.frame'. R package version 1.12.8. https://CRAN.R-project.org/package=data.table

Examples

```
sh.tar <- system.file('extdata/shinyApp/data/target_arab.txt', package='spatialHeatmap')
target.sh <- read_fr(sh.tar); target.sh[60:63, ]
```

`read_svg` *Parsing annotated SVG (aSVG) files*

Description

Parse one or multiple aSVG files and store their coordinates and related attributes in an `SVG` container, which will be used for creating spatial heatmap (SHM) plots.

Usage

```
read_svg(svg.path, raster.path = NULL, cores = 1, srsc = FALSE)
```

Arguments

`svg.path`	A vector of one or multiple paths of aSVG files. If multiple aSVGs, such as aSVGs depicting organs development across mutiple times, the aSVGs should be indexed with suffixes "_shm1", "_shm2", ..., such as "arabidopsis.thaliana_organ_shm1.svg", "arabidopsis.thaliana_organ_shm2.svg".
`raster.path`	• A vector of one or multiple paths of raster images in form of jpg or png, which are usually used as templates for creating aSVG images in `svg.path`. Optional (default is `NULL`), only applicable when superimposing raster images with SHM plots that are created from aSVG images. • Matching raster and aSVG images is indicated by identical base names such as imageA.png and imageA.svg. The layout order in SHMs composed of multiple independent images is controlled by numbering the corresponding file pairs accordingly such as imageA_1.png and imageA_1.svg, imageB_2.png and imageB_2.svg, etc.
`cores`	Number of CPUs to parse the aSVG files (default is `1`).
`srsc`	Logical. If 'TRUE', the aSVG is considered for co-visualizing spatially resolved single-cell and bulk data, and the rotation angle of the tissue section for spatial assays will be recorded.

Value

An object of `SVG` class, containing one or multiple aSVG instances.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Hadley Wickham, Jim Hester and Jeroen Ooms (2019). xml2: Parse XML. R package version 1.2.2. https://CRAN.R-project.org/package=xml2

See Also

SVG: the SVG class.

Examples

```
# The first raste image used as a template to create an aSVG.
raster.pa1 <- system.file('extdata/shinyApp/data/maize_leaf_shm1.png',
package='spatialHeatmap')
# The first aSVG created with the first template.
svg.pa1 <- system.file('extdata/shinyApp/data/maize_leaf_shm1.svg',
package='spatialHeatmap')
# The second raster image used as a template to create an aSVG.
raster.pa2 <- system.file('extdata/shinyApp/data/maize_leaf_shm2.png',
package='spatialHeatmap')
# The second aSVG created with the second template.
svg.pa2 <- system.file('extdata/shinyApp/data/maize_leaf_shm2.svg',
package='spatialHeatmap')

# Parse these two aSVGs without association with raster images.
svgs <- read_svg(svg.path=c(svg.pa1, svg.pa2), raster.path=NULL)
# Parse these two aSVGs. The raster image paths are provide so as to
# be associated with respective aSVGs, which will be used when
# superimposing raster images with SHM plots.
svgs <- read_svg(svg.path=c(svg.pa1, svg.pa2), raster.path=c(raster.pa1, raster.pa2))
```

---

reduce_dim *Reducing dimensionality in count data*

Description

A meta function for reducing dimensionality in count data.

Usage

```
reduce_dim(
  sce,
  choose.hvg = list(assay.type = "logcounts", block = NULL, more.var.args =
   list(transform = TRUE), top = 4000, more.choose.args = list(keep.ties = TRUE)),
  min.dim = 11,
```

```
  max.dim = 50,
  pca.arg = list(assay.type = "logcounts", block = NULL),
  umap.arg = list(num.dim = 2, reddim.type = "PCA"),
  tsne.arg = list(perplexity = 30, reddim.type = "PCA", output.name = "TSNE"),
  pca = FALSE
)
```

Arguments

sce	Normalized single cell data in SingleCellExperiment returned by norm_cell. Alternative forms include dgCMatrix, matrix, data.frame.
choose.hvg	Arguments in a named list passed to chooseRnaHvgs.se.
min.dim, max.dim	Integer scalars specifying the minimum (min.dim) and maximum (max.dim) number of (principle components) PCs to retain respectively in runPca.se.
pca.arg	Arguments in a named list passed to runPca.se.
umap.arg	Arguments in a named list passed to runUmap.se.
tsne.arg	Arguments in a named list passed to runTsne.se.
pca	Logical, if TRUE only the data with reduced dimentionality of PCA is returned. The default is FALSE, and UMAP and TSNE are also returned.

Value

A SingleCellExperiment object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. Lun ATL, Kancherla J (2023). "Powering single-cell analyses in the browser with WebAssembly." _Journal of Open Source Software_, *8*(89), 5603. doi:10.21105/joss.05603 <https://doi.org/10.21105/joss.05603>.

Examples

```
library(SingleCellExperiment)
set.seed(123)  # for reproducibility
# random counts: 1000 genes × 100 cells
counts_mat <- matrix(
  rpois(1000 * 100, lambda = 5), nrow = 1000, ncol = 100,
  dimnames = list(
    paste0("Gene", 1:1000),
```

```
    paste0("Cell", 1:100)
  )
)
sce <- SingleCellExperiment(
  assays = list(counts = counts_mat)
)
sce.qc = qc_cell(sce)
sce.norm <- norm_cell(sce.qc)
sce.dimred <- reduce_dim(sce.norm)
```

`reduce_rep` *Reduce sample replicates*

Description

In an expression profile matrix such as RNA-seq count table, where columns and rows are samples and biological molecules respectively, reduce sample replicates according to sum of correlation coefficients (Pearson, Spearman, Kendall).

Usage

```
reduce_rep(dat, n = 3, sim.meth = "pearson")
```

Arguments

`dat`	Abundance matrix in form of `data.frame` or `matrix`, where columns and rows are samples and biological molecules respectively. For example, gene exression matrix generated in RNA-seq.
`n`	An integer, the max number of replicates to keep per sample (e.g. tissue type). Within each sample, pairwise correlations are calculated among all replicates, and the correlations between one replicate and other relicates are summed. The replicates with top n largest sums are retained in each sample.
`sim.meth`	One of `pearson` (default), `kendall`, or `spearman`, indicating which correlation coefficient method to use for calculating similarities between replicates.

Value

A `matrix`.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

Examples

```
# Random abundance matrix.
dat <- matrix(rnorm(100), nrow=10)
# Two samples, each has 5 replicates.
colnames(dat) <- c(rep('sampleA', 5), rep('sampleB', 5))
rownames(dat) <- paste0('gene', seq_len(nrow(dat)))
reduce_rep(dat)
```

`return_feature` *Return aSVG Files Relevant to Target Features*

Description

This function parses a collection of aSVG files and returns those containing target features in a data frame. Successful spatial heatmap plotting requires the aSVG features of interest have matching samples (cells, tissues, *etc*) in the data. To meet this requirement, the returned features could be used to replace target sample counterparts in the data. Alternatively, the target samples in the data could be used to replace matching features in the aSVG through function `update_feature`. Refer to function `spatial_hm` for more details on aSVG files.

Usage

```
return_feature(
  feature,
  species,
  keywords.any = TRUE,
  remote = NULL,
  dir = NULL,
  svg.path = NULL,
  desc = FALSE,
  match.only = TRUE,
  return.all = FALSE
)
```

Arguments

`feature`	A vector of target feature keywords (case insentitive), which is used to select aSVG files from a collection. *E.g.* c('heart', 'brain'). If NA or NULL, all features of all SVG files matching `species` are returned.
`species`	A vector of target species keywords (case insentitive), which is used to select aSVG files from a collection. *E.g.* c('gallus'). If NA or NULL, all SVG files in `dir` are queried.
`keywords.any`	Logical, TRUE or FALSE. Default is TRUE. The internal searching is case-insensitive. The space, dot, hypen, semicolon, comma, forward slash are treated as separators between words and not counted in searching. If TRUE, every returned hit contains at least one word in the `feature` vector and at least one word

in the species vector, which means all the possible hits are returned. *E.g.* "prefrontal cortex" in "homo_sapiens.brain.svg" would be returned if feature=c('frontal') and species=c('homo'). If FALSE, every returned hit contains at least one exact element in the feature vector and all exact elements in the species vector. *E.g.* "frontal cortex" rather than "prefrontal cortex" in "homo_sapiens.brain.svg" would be returned if feature=c('frontal cortex') and species=c('homo sapiens', 'brain').

remote — Logical, FALSE or TRUE. If TRUE (default), the remote EBI aSVG repository https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg and spatialHeatmap aSVG Repository https://github.com/jianhaizhang/SHM_SVG/tree/master/SHM_SVG_repo developed in this project are queried.

dir — The directory path of aSVG files. If remote is TRUE, the returned aSVG files are saved in this directory. Note existing aSVG files with same names as returned ones are overwritten. If remote is FALSE, user-provided (local) aSVG files should be saved in this directory for query. Default is NULL.

svg.path — The path of a specific aSVG file. If the provided aSVG file exists, only features of this file are returned and there will be no querying process. Default is NULL.

desc — Logical, FALSE or TRUE. Default is FALSE. If TRUE, the feature descriptions from the R package "rols" (Laurent Gatto 2019) are added. If too many features are returned, this process takes a long time.

match.only — Logical, TRUE or FALSE. If TRUE (default), only target features are returned. If FALSE, all features in the matching aSVG files are returned, and the matching features are moved on the top of the data frame.

return.all — Logical, FALSE or TRUE. Default is FALSE. If TRUE, all features together with all respective aSVG files are returned, regardless of feature and species.

Value

A data frame containing information on target features and aSVGs.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Laurent Gatto (2019). rols: An R interface to the Ontology Lookup Service. R package version 2.14.0. http://lgatto.github.com/rols/

Hadley Wickham, Jim Hester and Jeroen Ooms (2019). xml2: Parse XML. R package version 1.2.2. https://CRAN.R-project.org/package=xml2

R Core Team (2019). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505-9

Examples

```
# This function is able to work on the EBI aSVG repository directly: https://github.com/
# ebi-gene-expression-group/anatomogram/tree/master/src/svg. The following shows how to
# download a chicken aSVG containing spatial features of 'brain' and 'heart'. An empty
# directory is recommended so as to avoid overwriting existing SVG files.
# Here "~/test" is used.


# Make an empty directory "~/test" if not exist.
if (!dir.exists('~/test')) dir.create('~/test')
# Remote aSVG repos.
data(aSVG.remote.repo)
tmp.dir <- normalizePath(tempdir(check=TRUE), winslash="/", mustWork=FALSE)
tmp.dir.ebi <- paste0(tmp.dir, '/ebi.zip')
tmp.dir.shm <- paste0(tmp.dir, '/shm.zip')
# Download the remote aSVG repos as zip files. According to Bioconductor's
# requirements, downloadings are not allowed inside functions, so the repos are
# downloaded before calling "return_feature".
download.file(aSVG.remote.repo$ebi, tmp.dir.ebi)
download.file(aSVG.remote.repo$shm, tmp.dir.shm)
remote <- list(tmp.dir.ebi, tmp.dir.shm)
# Query the remote aSVG repos.
feature.df <- return_feature(feature=c('heart', 'brain'), species=c('gallus'), dir='~/test',
match.only=FALSE, remote=remote)
feature.df
# The path of downloaded aSVG.
svg.chk <- '~/test/gallus_gallus.svg'


# The spatialHeatmap package has a small aSVG collection and can be used to demonstrate the
# local query.
# Get the path of local aSVGs from the package.
svg.dir <- system.file("extdata/shinyApp/data", package="spatialHeatmap")
# Query the local aSVG repo. The "species" argument is set NULL on purpose so as to illustrate
# how to select the target aSVG among all matching aSVGs.
feature.df <- return_feature(feature=c('heart', 'brain'), species=NULL, dir=svg.dir,
match.only=FALSE, remote=NULL)
# All matching aSVGs.
unique(feature.df$SVG)
# Select the target aSVG of chicken.
subset(feature.df, SVG=='gallus_gallus.svg')
```

---

save_cache *Save R Objects in Cache*

Description

Save R Objects in Cache

Usage

```
save_cache(dir = NULL, overwrite = TRUE, obj, na = NULL)
```

Arguments

dir	The directory path to save the cached data. Default is NULL and the cached data is stored in ~/.cache/shm.
overwrite	Logical, TRUE or FALSE. Default is TRUE and data in the cache with the same name of the object in ... will be overwritten.
obj, na	A single R object ('obj') to be cached into the file named 'na'.

Value

The directory path of the cache.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Lori Shepherd and Martin Morgan (2020). BiocFileCache: Manage Files Across Sessions. R package version 1.12.1.

Examples

```
# Save the object "iris" in the default cache "~/.cache/shm".
cache.pa <- save_cache(dir=NULL, overwrite=TRUE, iris)
```

---

shiny_shm *Integrated Shiny App*

Description

In additon to generating spatial heatmaps and corresponding item (genes, proteins, metabolites, *etc.*) context plots from R, spatialHeatmap includes a Shiny App (https://shiny.rstudio.com/) that provides access to the same functionalities from an intuitive-to-use web browser interface. Apart from being very user-friendly, this App conveniently organizes the results of the entire visualization workflow in a single browser window with options to adjust the parameters of the individual components interactively. Upon launched, the app automatically displays a pre-formatted example. To use this app, the data matrix (*e.g.* gene expression matrix) and aSVG image are uploaded as tabular text (*e.g.* in CSV or TSV format) and SVG file, respectively. To also allow users to upload data matrix stored in SummarizedExperiment objects, one can export them from R to a tabular file with the filter_data function. In this function call, the user sets a desired directory path under dir. Within this directory the tabular file will be written to "customComputedData/sub_matrix.txt" in TSV format. The column names in the exported tabular file preserve the experimental design information

from the colData slot by concatenating the corresponding sample and condition information separated by double underscores. To interactively view functional descriptions by moving the cursor over network nodes, the corresponding annotation column needs to be present in the rowData slot and its column name assigned to the ann argument. In the exported tabular file the extra annotation column is appended to the expression matrix. See function filter_data for details. If the subsetted data matrix in the Matrix Heatmap is too large, *e.g.* >10,000 rows, the "customComputedData" under "Step 1: data sets" is recommended. Since this subsetted matrix is fed to the Network, and the internal computation of adjacency matrix and module identification would be intensive. In order to protect the app from crash, the intensive computation should be performed outside the app, then upload the results under "customComputedData". When using "customComputedData", the data matrix to upload is the subsetted matrix "sub_matrix.txt" generated with submatrix, which is a TSV-tabular text file. The adjacency matrix and module assignment to upload are "adj.txt" and "mod.txt" generated in function adj_mod respectively. Note, "sub_matrix.txt", "adj.txt", and "mod.txt" are downstream to the same call on filter_data, so the three files should not be mixed between different filtering when uploading. See the instruction page in the app for details. The large matrix issue could be resolved by increasing the subsetting strigency to get smaller matrix in submatrix in most cases. Only in rare cases users cannot avoid very large subsetted matrix, the "customComputedData" is recommended.

Usage

```
shiny_shm()
```

Value

A web browser based Shiny app.

Details

No argument is required, this function launches the Shiny app directly.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

https://www.w3schools.com/graphics/svg_intro.asp

https://shiny.rstudio.com/tutorial/

https://shiny.rstudio.com/articles/datatables.html

https://rstudio.github.io/DT/010-style.html

https://plot.ly/r/heatmaps/

https://www.gimp.org/tutorials/

https://inkscape.org/en/doc/tutorials/advanced/tutorial-advanced.en.html

http://www.microugly.com/inkscape-quickguide/

https://cran.r-project.org/web/packages/visNetwork/vignettes/Introduction-to-visNetwork.html

Winston Chang, Joe Cheng, JJ Allaire, Yihui Xie and Jonathan McPherson (2017). shiny: Web Application Framework for R. R package version 1.0.3. https://CRAN.R-project.org/package=shiny

Winston Chang and Barbara Borges Ribeiro (2017). shinydashboard: Create Dashboards with 'Shiny'. R package version 0.6.1. https://CRAN.R-project.org/package=shinydashboard

Paul Murrell (2009). Importing Vector Graphics: The grImport Package for R. Journal of Statistical Software, 30(4), 1-37. URL http://www.jstatsoft.org/v30/i04/

Jeroen Ooms (2017). rsvg: Render SVG Images into PDF, PNG, PostScript, or Bitmap Arrays. R package version 1.1. https://CRAN.R-project.org/package=rsvg

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Yihui Xie (2016). DT: A Wrapper of the JavaScript Library 'DataTables'. R package version 0.2. https://CRAN.R-project.org/package=DT

Baptiste Auguie (2016). gridExtra: Miscellaneous Functions for "Grid" Graphics. R package version 2.2.1. https://CRAN.R-project.org/package=gridExtra

Andrie de Vries and Brian D. Ripley (2016). ggdendro: Create Dendrograms and Tree Diagrams Using 'ggplot2'. R package version 0.1-20. https://CRAN.R-project.org/package=ggdendro

Langfelder P and Horvath S, WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics 2008, 9:559 doi:10.1186/1471-2105-9-559

Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/

Simon Urbanek and Jeffrey Horner (2015). Cairo: R graphics device using cairo graphics library for creating high-quality bitmap (PNG, JPEG, TIFF), vector (PDF, SVG, PostScript) and display (X11 and Win32) output. R package version 1.5-9. https://CRAN.R-project.org/package=Cairo

R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Duncan Temple Lang and the CRAN Team (2017). XML: Tools for Parsing and Generating XML Within R and S-Plus. R package version 3.98-1.9. https://CRAN.R-project.org/package=XML

Carson Sievert, Chris Parmer, Toby Hocking, Scott Chamberlain, Karthik Ram, Marianne Corvellec and Pedro Despouy (NA). plotly: Create Interactive Web Graphics via 'plotly.js'. https://plot.ly/r, https://cpsievert.github.io/plotly_book/, https://github.com/ropensci/plotly

Matt Dowle and Arun Srinivasan (2017). data.table: Extension of 'data.frame'. R package version 1.10.4. https://CRAN.R-project.org/package=data.table

R. Gentleman, V. Carey, W. Huber and F. Hahne (2017). genefilter: genefilter: methods for filtering genes from high-throughput experiments. R package version 1.58.1.

Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/

Almende B.V., Benoit Thieurmel and Titouan Robert (2017). visNetwork: Network Visualization using 'vis.js' Library. R package version 2.0.1. https://CRAN.R-project.org/package=visNetwork Zhang L (2023). _spsComps: 'systemPipeShiny' UI and Server Components_. R package version 0.3.3.0, <https://CRAN.R-project.org/package=spsComps>.

Examples

```
shiny_shm()
```

shm *Plot Spatial Heatmaps*

Description

The input are a pair of annotated SVG (aSVG) file and formatted numeric data (`vector`, `data.frame`, `SummarizedExperiment`). In the aSVG, spatial features (tissues, organs, etc) are represented by shapes and assigned unique identifiers, and the numeric data are measured from these spatial features. In biological applications, aSVGs are anatomical structures, and numeric data are expression values of genes, proteins, metabolites, etc assayed from spatial features. Numeric data are mapped to spatial features in aSVGs according to the same identifiers between the two. The mapped features are filled with colors translated from the numeric data while other features are transparent. The output images are termed spatial heatmaps (SHMs).

This function is designed multi-functional for maximal flexibility. For example, SHMs can be organized horizontally by each gene or each experimental variable to facilitate comparisons. In multi-layer anotomical structures, selected tissues can be set transparent to expose burried features in lower layers. The color scale is customizable to highlight difference across features. This function also works with many other types of spatial data, such as population data plotted to geographic maps.

Usage

```
## S4 method for signature 'SPHM'
shm(
  data,
  assay.na = NULL,
  sam.factor = NULL,
  con.factor = NULL,
  ID,
  charcoal = FALSE,
  alpha.overlay = 1,
  lay.shm = "gene",
  ncol = 2,
  h = 0.99,
  col.com = c("yellow", "orange", "red"),
  col.bar = "selected",
  thr = c(NA, NA),
  cores = NA,
  bar.width = 0.08,
  bar.title = NULL,
  bar.title.size = 0,
  scale = "no",
  ft.trans = NULL,
  tis.trans = ft.trans,
  legend.r = 0.9,
  sub.title.size = 11,
```

```
  sub.title.vjust = 2,
  legend.plot = "all",
  ft.legend = "identical",
  bar.value.size = 10,
  legend.plot.title = "Legend",
  legend.plot.title.size = 11,
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.position = "bottom",
  legend.direction = NULL,
  legend.key.size = 0.02,
  legend.text.size = 12,
  angle.text.key = NULL,
  position.text.key = NULL,
  legend.2nd = FALSE,
  position.2nd = "bottom",
  legend.nrow.2nd = NULL,
  legend.ncol.2nd = NULL,
  legend.key.size.2nd = 0.03,
  legend.text.size.2nd = 10,
  angle.text.key.2nd = 0,
  position.text.key.2nd = "right",
  add.feature.2nd = FALSE,
  label = FALSE,
  label.size = 4,
  label.angle = 0,
  hjust = 0,
  vjust = 0,
  opacity = 1,
  key = TRUE,
  line.width = 0.2,
  image = NULL,
  line.color = "grey70",
  relative.scale = NULL,
  out.dir = NULL,
  animation.scale = 1,
  aspr = 1,
  selfcontained = FALSE,
  video.dim = "640x480",
  res = 500,
  interval = 1,
  framerate = 1,
  bar.width.vdo = 0.1,
  legend.value.vdo = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

data	An 'SHM' class that containing the numeric data and aSVG instances for plotting SHMs or co-visualization plots. See SPHM.
assay.na	The name of target assay to use when data is SummarizedExperiment.
sam.factor	The column name corresponding to spatial features in colData of SummarizedExperiment. If the column names in the assay slot already follows the scheme "spatialFeature__variable", then the colData slot is not required and accordingly this argument could be NULL.
con.factor	The column name corresponding to experimental variables in colData of SummarizedExperiment. It can be NULL if column names of in the assay slot already follows the scheme "spatialFeature__variable", or no variable is associated with the data.
ID	A character vector of assyed items (*e.g.* genes, proteins) whose abudance values are used to color the aSVG.
charcoal	Logical, if TRUE the raster image will be turned black and white.
alpha.overlay	The opacity of the raster image under the SHM when superimposing raster images with SHMs. The default is 1.
lay.shm	One of 'gene', 'con', or 'none'. If 'gene', SHMs are organized horizontally by each biomolecule (gene, protein, or metabolite, *etc.*) and variables are sorted under each biomolecule. If 'con', SHMs are organized horizontally by each experiment vairable and biomolecules are sorted under each variable. If 'none', SHMs are organized by the biomolecule order in ID and variables follow the order they appear in data.
ncol	The number of columns to display SHMs, which does not include the legend plot.
h	The height (0-1) of color key and SHM/co-visualization plots in the middle, not including legend plots.
col.com	A vector of color components used to build the color scale. The default is 'c('yellow', 'orange', 'red')'.
col.bar	One of 'selected' or 'all', the former uses expression values of ID to build the color scale while the latter uses all expression values from the data. The default is 'selected'.
thr	A two-numeric vector of expression value thresholds (the range of the color bar). The first and the second element will be the minmun and maximum threshold in the color bar respectively. Values above the max or below min will be assigned the same color as the max or min respectively. The default is c(NA, NA) and the min and max values in the data will be used. If one needs to change only max or min, the other should be NA.
cores	The number of CPU cores for parallelization. The default is 'NA', and the number of used cores is 1 or 2 depending on the availability.
bar.width	The width of color bar that ranges from 0 to 1. The default is 0.08.
bar.title, bar.title.size	The title and title size of the color key.

`scale` One of `no` (default), `selected`, `all`, or `row`, corresponding to no scaling, scaling selected rows as a whole, scaling all rows as a whole, or scaling each row independently.

`ft.trans` A character vector of spatial features that will be set transparent. When features of interest are covered by overlapping features on the top layers and the latter can be set transparent.

`tis.trans` This argument is deprecated and replaced by `ft.trans`.

`legend.r` A numeric (-1 to 1) to adjust the legend plot size.

`sub.title.size` A numeric of the subtitle font size of each individual spatial heatmap. The default is 11.

`sub.title.vjust`
A numeric of vertical adjustment for subtitle. The default is 2.

`legend.plot` A vector of suffix(es) of aSVG file name(s) such as `c('shm1', 'shm2')`. Only aSVG(s) whose suffix(es) are assigned to this arugment will have a legend plot on the right. The default is `all` and each aSVG will have a legend plot. If NULL, no legend plot is shown.

`ft.legend` One of "identical", "all", or a character vector of tissue/spatial feature identifiers from the aSVG file. The default is "identical" and all the identical/matching tissues/spatial features between the data and aSVG file are colored in the legend plot. If "all", all tissues/spatial features in the aSVG are shown. If a vector, only the tissues/spatial features in the vector are shown.

`bar.value.size` A numeric of value size in the y-axis of the color bar. The default is 10.

`legend.plot.title`
The title of the legend plot. The default is 'Legend'.

`legend.plot.title.size`
The title size of the legend plot. The default is 11.

`legend.ncol` An integer of the total columns of keys in the legend plot. The default is NULL. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of shown spatial features.

`legend.nrow` An integer of the total rows of keys in the legend plot. The default is NULL. It is only applicable to the legend plot. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of matching spatial features.

`legend.position`
the default position of legends ("none", "left", "right", "bottom", "top", "inside")

`legend.direction`
layout of items in legends ("horizontal" or "vertical")

`legend.key.size`
A numeric of the legend key size ("npc"), applicable to the legend plot. The default is 0.02.

`legend.text.size`
A numeric of the legend label size, applicable to the legend plot. The default is 12.

`angle.text.key` Key text angle in legend plots. The default is NULL, equivalent to 0.

`position.text.key`
: The position of key text in legend plots, one of 'top', 'right', 'bottom', 'left'. Default is NULL, equivalent to 'right'.

`legend.2nd`
: Logical. If 'TRUE', the secondary legend is added to each SHM, which are the numeric values of each colored spatial features. The default its 'FALSE'. Only applies to the static image.

`position.2nd`
: The position of the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left', or a two-component numeric vector. The default is 'bottom'. Applies to images and videos.

`legend.nrow.2nd`
: An integer of rows of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.ncol.2nd`
: An integer of columns of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.key.size.2nd`
: A numeric of legend key size in SHMs. The default is 0.03. Applies to static images and videos.

`legend.text.size.2nd`
: A numeric of the secondary legend text size in SHMs. The default is 10. Applies to static images and videos.

`angle.text.key.2nd`
: Angle of the key text in the secondary legend in SHMs. Default is 0. Applies to static images and videos.

`position.text.key.2nd`
: The position of key text in the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left'. Default is 'right'. Applies to static images and videos.

`add.feature.2nd`
: Logical. If 'TRUE', feature identifiers are added to the secondary legend. The default is FALSE. Applies to static images of SHMs.

`label`
: Logical. If 'TRUE', the same spatial features between numeric data and aSVG are labeled by their identifiers. The default is 'FALSE'. It is useful when spatial features are labeled by similar colors.

`label.size`
: The size of spatial feature labels in legend plots. The default is 4.

`label.angle`
: The angle of spatial feature labels in legend plots. Default is 0.

`hjust`, `vjust`
: The value to horizontally or vertically adjust positions of spatial feature labels in legend plots respectively. Default of both is 0.

`opacity`
: The transparency of colored spatial features in legend plots. Default is 1. If 0, features are totally transparent.

`key`
: Logical. If 'TRUE' (default), keys are added in legend plots. If `label` is TRUE, the keys could be removed.

`line.width`
: The thickness of each shape outline in the aSVG is maintained in spatial heatmaps, *i.e.* the stroke widths in Inkscape. This argument is the extra thickness added to all outlines. Default is 0.2 in case stroke widths in the aSVG are 0.

image	Name of SpatialImage object to get coordinates for. If NULL (default), will retrieve coordinates for the default image. See GetTissueCoordinates in SeuratObject for details.
line.color	A character of the shape outline color. Default is "grey70".
relative.scale	A numeric to adjust the relative sizes between multiple aSVGs. Applicable only if multiple aSVGs are provided. Default is NULL and all aSVGs have the same size.
out.dir	The directory to save SHMs as interactive HTML files and videos. Default is 'NULL', and the HTML files and videos are not saved.
animation.scale	A numeric to scale the SHM size in the HTML files. The default is 1, and the height is 550px and the width is calculated according to the original aspect ratio in the aSVG file.
aspr	The aspect ratio (width to height) in the interative HTML files.
selfcontained	Whether to save the HTML as a single self-contained file (with external resources base64 encoded) or a file with external resources placed in an adjacent directory.
video.dim	A single character of the dimension of video frame in form of 'widthxheight', such as '1920x1080', '1280x800', '320x568', '1280x1024', '1280x720', '320x480', '480x360', '600x600', '800x600', '640x480' (default). The aspect ratio of SHMs are decided by width and height.
res	Resolution of the video in dpi.
interval	The time interval (seconds) between SHM frames in the video. Default is 1.
framerate	An integer of video framerate in frames per seconds. Default is 1. Larger values make the video smoother.
bar.width.vdo	The color bar width (0-1) in videos.
legend.value.vdo	Logical. If 'TRUE', numeric values of colored spatial features are added to the video legend. The default is NULL.
verbose	Logical. If 'TRUE' (default), intermediate messages will be printed.
...	additional element specifications not part of base ggplot2. In general, these should also be defined in the element tree argument. Splicing a list is also supported.

Value

An 'SPHM' object.

Details

See the package vignette (browseVignettes('spatialHeatmap')).

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

https://www.gimp.org/tutorials/ https://inkscape.org/en/doc/tutorials/advanced/tutorial-advanced.en.html http://www.microugly.com/inkscape-quickguide/ Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1 H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016. Jeroen Ooms (2018). rsvg: Render SVG Images into PDF, PNG, PostScript, or Bitmap Arrays. R package version 1.3. https://CRAN.R-project.org/package=rsvg R. Gentleman, V. Carey, W. Huber and F. Hahne (2017). genefilter: genefilter: methods for filtering genes from high-throughput experiments. R package version 1.58.1 Paul Murrell (2009). Importing Vector Graphics: The grImport Package for R. Journal of Statistical Software, 30(4), 1-37. URL http://www.jstatsoft.org/v30/i04/ Baptiste Auguie (2017). gridExtra: Miscellaneous Functions for "Grid" Graphics. R package version 2.3. https://CRAN.R-project.org/package=gridExtra R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. RL https://www.R-project.org/ https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg Yu, G., 2020. ggplotify: Convert Plot to 'grob' or 'ggplot' Object. R package version 0.0.5.URLhttps://CRAN.R-project.org/package=ggplotify30 Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 Guangchuang Yu (2020). ggplotify: Convert Plot to 'grob' or 'ggplot' Object. R package version 0.0.5. https://CRAN.R-project.org/package=ggplotify Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Marques A et al. (2016). Oligodendrocyte heterogeneity in the mouse juvenile and adult central nervous system. Science 352(6291), 1326-1329. Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x. Vacher, Claire-Marie, Helene Lacaille, Jiaqi J O'Reilly, Jacquelyn Salzbank, Dana Bakalar, Sonia Sebaoui, Philippe Liere, et al. 2021. "Placental Endocrine Function Shapes Cerebellar Development and Social Behavior." Nat. Neurosci. 24 (10). Nature Publishing Group: 1392–1401

Examples

```
## The example data included in this package come from an RNA-seq analysis on
## development of 7 chicken organs under 9 time points (Cardoso-Moreira et al. 2019).
## The complete raw count data are downloaded using the R package ExpressionAtlas
## (Keays 2019) with the accession number "E-MTAB-6769".

# Access example count data.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is made based on the
# experiment design.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
```

```
target.chk[1:5, ]
# Store example data in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)

# Normalize data.
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)

# The chicken aSVG downloaded from the EBI aSVG repository (https://github.com/ebi-gene-
# expression-group/anatomogram/tree/master/src/svg) is included in this package and
# accessed as below.
svg.chk <- system.file("extdata/shinyApp/data", "gallus_gallus.svg",
package="spatialHeatmap")
# Read the chicken aSVG file.
svg.chk <- read_svg(svg.path=svg.chk)

# Store assay data and aSVG in an "SHM" class.
dat.chk <- SPHM(svg=svg.chk, bulk=se.chk.fil)
# Plot spatial heatmaps with gene "ENSGALG00000019846".
shm(data=dat.chk, ID='ENSGALG00000019846', legend.r=1.9,
legend.nrow=5, sub.title.size=7, ncol=3)

# Save SHMs as HTML and video files in the "~/test" directory.

if (!dir.exists('~/test')) dir.create('~/test')
shm(data=dat.chk, ID='ENSGALG00000019846', legend.r=1.9,
legend.nrow=5, sub.title.size=7, ncol=3, out.dir='~/test')
```

`spatial_hm` *Plot Spatial Heatmaps*

Description

The input are a pair of annotated SVG (aSVG) file and formatted data (`vector`, `data.frame`, `SummarizedExperiment`). In the former, spatial features are represented by shapes and assigned unique identifiers, while the latter are numeric values measured from these spatial features and organized in specific formats. In biological cases, aSVGs are anatomical or cell structures, and data

are measurements of genes, proteins, metabolites, *etc.* in different samples (*e.g.* cells, tissues). Data are mapped to the aSVG according to identifiers of assay samples and aSVG features. Only the data from samples having matching counterparts in aSVG features are mapped. The mapped features are filled with colors translated from the data, and the resulting images are termed spatial heatmaps. Note, "sample" and "feature" are two equivalent terms referring to cells, tissues, organs *etc.* where numeric values are measured. Matching means a target sample in data and a target spatial feature in aSVG have the same identifier.

This function is designed as much flexible as to achieve optimal visualization. For example, subplots of spatial heatmaps can be organized by gene or condition for easy comparison, in multi-layer anotomical structures selected tissues can be set transparent to expose burried features, color scale is customizable to highlight difference among features. This function also works with many other types of spatial data, such as population data plotted to geographic maps.

Usage

```
## S4 method for signature 'SVG'
spatial_hm(
  svg,
  data,
  assay.na = NULL,
  sam.factor = NULL,
  con.factor = NULL,
  ID,
  charcoal = FALSE,
  alpha.overlay = 1,
  lay.shm = "gene",
  ncol = 2,
  col.com = c("yellow", "orange", "red"),
  col.bar = "selected",
  thr = c(NA, NA),
  cores = NA,
  bar.width = 0.08,
  bar.title = NULL,
  bar.title.size = 0,
  scale = NULL,
  ft.trans = NULL,
  tis.trans = ft.trans,
  lis.rematch = NULL,
  legend.r = 0.9,
  image = NULL,
  sub.title.size = 11,
  sub.title.vjust = 2,
  legend.plot = "all",
  ft.legend = "identical",
  bar.value.size = 10,
  legend.plot.title = "Legend",
  legend.plot.title.size = 11,
  legend.ncol = NULL,
  legend.nrow = NULL,
```

```
  legend.position = "bottom",
  legend.direction = NULL,
  legend.key.size = 0.02,
  legend.text.size = 12,
  angle.text.key = NULL,
  position.text.key = NULL,
  legend.2nd = FALSE,
  position.2nd = "bottom",
  legend.nrow.2nd = NULL,
  legend.ncol.2nd = NULL,
  legend.key.size.2nd = 0.03,
  legend.text.size.2nd = 10,
  angle.text.key.2nd = 0,
  position.text.key.2nd = "right",
  add.feature.2nd = FALSE,
  label = FALSE,
  label.size = 4,
  label.angle = 0,
  hjust = 0,
  vjust = 0,
  opacity = 1,
  key = TRUE,
  line.width = 0.2,
  line.color = "grey70",
  relative.scale = NULL,
  verbose = TRUE,
  out.dir = NULL,
  animation.scale = 1,
  selfcontained = FALSE,
  video.dim = "640x480",
  res = 500,
  interval = 1,
  framerate = 1,
  bar.width.vdo = 0.1,
  legend.value.vdo = NULL,
  ...
)
```

Arguments

svg	An SVG object containing one or multiple aSVG instances (see SVG and read_svg). In the aSVGs, spatial features (tissues, organs, etc) having counterparts with the same identifiers in the 'bulk' data will be colored accoording to expression profiles of chosen biomolecules (genes, proteins, etc).
data	An 'SHM' class that containing the numeric data and aSVG instances for plotting SHMs or co-visualization plots. See SPHM.
assay.na	The name of target assay to use when data is SummarizedExperiment.
sam.factor	The column name corresponding to spatial features in colData of SummarizedExperiment.

	If the column names in the assay slot already follows the scheme "spatialFeature__variable", then the colData slot is not required and accordingly this argument could be NULL.
con.factor	The column name corresponding to experimental variables in colData of SummarizedExperiment. It can be NULL if column names of in the assay slot already follows the scheme "spatialFeature__variable", or no variable is associated with the data.
ID	A character vector of assyed items (*e.g.* genes, proteins) whose abudance values are used to color the aSVG.
charcoal	Logical, if TRUE the raster image will be turned black and white.
alpha.overlay	The opacity of the raster image under the SHM when superimposing raster images with SHMs. The default is 1.
lay.shm	One of 'gene', 'con', or 'none'. If 'gene', SHMs are organized horizontally by each biomolecule (gene, protein, or metabolite, *etc.*) and variables are sorted under each biomolecule. If 'con', SHMs are organized horizontally by each experiment vairable and biomolecules are sorted under each variable. If 'none', SHMs are organized by the biomolecule order in ID and variables follow the order they appear in data.
ncol	The number of columns to display SHMs, which does not include the legend plot.
col.com	A vector of color components used to build the color scale. The default is 'c('yellow', 'orange', 'red')'.
col.bar	One of 'selected' or 'all', the former uses expression values of ID to build the color scale while the latter uses all expression values from the data. The default is 'selected'.
thr	A two-numeric vector of expression value thresholds (the range of the color bar). The first and the second element will be the minmun and maximum threshold in the color bar respectively. Values above the max or below min will be assigned the same color as the max or min respectively. The default is c(NA, NA) and the min and max values in the data will be used. If one needs to change only max or min, the other should be NA.
cores	The number of CPU cores for parallelization. The default is 'NA', and the number of used cores is 1 or 2 depending on the availability.
bar.width	The width of color bar that ranges from 0 to 1. The default is 0.08.
bar.title, bar.title.size	The title and title size of the color key.
scale	One of no (default), selected, all, or row, corresponding to no scaling, scaling selected rows as a whole, scaling all rows as a whole, or scaling each row independently.
ft.trans	A character vector of spatial features that will be set transparent. When features of interest are covered by overlapping features on the top layers and the latter can be set transparent.
tis.trans	This argument is deprecated and replaced by ft.trans.
lis.rematch	The list for re-matching in SHMs (only bulk data) or matching between single-cell and bulk data in co-visualization.

SHMs A named `list` for rematching spatial features between numeric data (ftA, ftB) and aSVGs (ftC, ftD, ftE). In each slot, the slot name is a spatial feature from the data and the corresponding element is one or multiple spatial features from the aSVG. *E.g.* `list(ftA = c('ftC', 'ftD'), ftB = c('ftE'))`.

Co-visualization plots Mapping cells to tissues: a named `list`, where cell group labels from `colData(sce. dimred)[, 'cell.group']` are the name slots and aSVG features are the corresponding `list` elements. Mapping tissues to cells: a named `list`, where tissues are the name slots and cells from `colData(sce.dimred)[, 'cell.group']` are the corresponding `list` elements. Applicable when cell grouping methods are annodation labels, marker genes, clustering, or manual assignments.

`legend.r` A numeric (-1 to 1) to adjust the legend plot size.

`image` Name of SpatialImage object to get coordinates for. If NULL (default), will retrieve coordinates for the default image. See `GetTissueCoordinates` in `SeuratObject` for details.

`sub.title.size` A numeric of the subtitle font size of each individual spatial heatmap. The default is 11.

`sub.title.vjust`
A numeric of vertical adjustment for subtitle. The default is `2`.

`legend.plot` A vector of suffix(es) of aSVG file name(s) such as `c('shm1', 'shm2')`. Only aSVG(s) whose suffix(es) are assigned to this arugment will have a legend plot on the right. The default is `all` and each aSVG will have a legend plot. If NULL, no legend plot is shown.

`ft.legend` One of "identical", "all", or a character vector of tissue/spatial feature identifiers from the aSVG file. The default is "identical" and all the identical/matching tissues/spatial features between the data and aSVG file are colored in the legend plot. If "all", all tissues/spatial features in the aSVG are shown. If a vector, only the tissues/spatial features in the vector are shown.

`bar.value.size` A numeric of value size in the y-axis of the color bar. The default is 10.

`legend.plot.title`
The title of the legend plot. The default is 'Legend'.

`legend.plot.title.size`
The title size of the legend plot. The default is 11.

`legend.ncol` An integer of the total columns of keys in the legend plot. The default is NULL. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of shown spatial features.

`legend.nrow` An integer of the total rows of keys in the legend plot. The default is NULL. It is only applicable to the legend plot. If both `legend.ncol` and `legend.nrow` are used, the product of the two arguments should be equal or larger than the total number of matching spatial features.

`legend.position`
the default position of legends ("none", "left", "right", "bottom", "top", "inside")

`legend.direction`
layout of items in legends ("horizontal" or "vertical")

`legend.key.size`
: A numeric of the legend key size ("npc"), applicable to the legend plot. The default is 0.02.

`legend.text.size`
: A numeric of the legend label size, applicable to the legend plot. The default is 12.

`angle.text.key` Key text angle in legend plots. The default is NULL, equivalent to 0.

`position.text.key`
: The position of key text in legend plots, one of 'top', 'right', 'bottom', 'left'. Default is NULL, equivalent to 'right'.

`legend.2nd` Logical. If 'TRUE', the secondary legend is added to each SHM, which are the numeric values of each colored spatial features. The default its 'FALSE'. Only applies to the static image.

`position.2nd` The position of the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left', or a two-component numeric vector. The default is 'bottom'. Applies to images and videos.

`legend.nrow.2nd`
: An integer of rows of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.ncol.2nd`
: An integer of columns of the secondary legend keys in SHMs. Applies to static images and videos.

`legend.key.size.2nd`
: A numeric of legend key size in SHMs. The default is 0.03. Applies to static images and videos.

`legend.text.size.2nd`
: A numeric of the secondary legend text size in SHMs. The default is 10. Applies to static images and videos.

`angle.text.key.2nd`
: Angle of the key text in the secondary legend in SHMs. Default is 0. Applies to static images and videos.

`position.text.key.2nd`
: The position of key text in the secondary legend in SHMs, one of 'top', 'right', 'bottom', 'left'. Default is 'right'. Applies to static images and videos.

`add.feature.2nd`
: Logical. If 'TRUE', feature identifiers are added to the secondary legend. The default is FALSE. Applies to static images of SHMs.

`label` Logical. If 'TRUE', the same spatial features between numeric data and aSVG are labeled by their identifiers. The default is 'FALSE'. It is useful when spatial features are labeled by similar colors.

`label.size` The size of spatial feature labels in legend plots. The default is 4.

`label.angle` The angle of spatial feature labels in legend plots. Default is 0.

`hjust`, `vjust` The value to horizontally or vertically adjust positions of spatial feature labels in legend plots respectively. Default of both is 0.

opacity	The transparency of colored spatial features in legend plots. Default is 1. If 0, features are totally transparent.
key	Logical. If 'TRUE' (default), keys are added in legend plots. If label is TRUE, the keys could be removed.
line.width	The thickness of each shape outline in the aSVG is maintained in spatial heatmaps, *i.e.* the stroke widths in Inkscape. This argument is the extra thickness added to all outlines. Default is 0.2 in case stroke widths in the aSVG are 0.
line.color	A character of the shape outline color. Default is "grey70".
relative.scale	A numeric to adjust the relative sizes between multiple aSVGs. Applicable only if multiple aSVGs are provided. Default is NULL and all aSVGs have the same size.
verbose	Logical. If 'TRUE' (default), intermediate messages will be printed.
out.dir	The directory to save SHMs as interactive HTML files and videos. Default is 'NULL', and the HTML files and videos are not saved.
animation.scale	A numeric to scale the SHM size in the HTML files. The default is 1, and the height is 550px and the width is calculated according to the original aspect ratio in the aSVG file.
selfcontained	Whether to save the HTML as a single self-contained file (with external resources base64 encoded) or a file with external resources placed in an adjacent directory.
video.dim	A single character of the dimension of video frame in form of 'widthxheight', such as '1920x1080', '1280x800', '320x568', '1280x1024', '1280x720', '320x480', '480x360', '600x600', '800x600', '640x480' (default). The aspect ratio of SHMs are decided by width and height.
res	Resolution of the video in dpi.
interval	The time interval (seconds) between SHM frames in the video. Default is 1.
framerate	An integer of video framerate in frames per seconds. Default is 1. Larger values make the video smoother.
bar.width.vdo	The color bar width (0-1) in videos.
legend.value.vdo	Logical. If 'TRUE', numeric values of colored spatial features are added to the video legend. The default is NULL.
...	additional element specifications not part of base ggplot2. In general, these should also be defined in the element tree argument. Splicing a list is also supported.

Value

An image of spatial heatmap(s), a two-component list of the spatial heatmap(s) in ggplot format and a data.frame of mapping between assayed samples and aSVG features.

Details

See the package vignette (browseVignettes('spatialHeatmap')).

Author(s)

Jianhai Zhang <jianhai.zhang@email.ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

https://www.gimp.org/tutorials/
https://inkscape.org/en/doc/tutorials/advanced/tutorial-advanced.en.html
http://www.microugly.com/inkscape-quickguide/ Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1
H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.
Jeroen Ooms (2018). rsvg: Render SVG Images into PDF, PNG, PostScript, or Bitmap Arrays. R package version 1.3. https://CRAN.R-project.org/package=rsvg
R. Gentleman, V. Carey, W. Huber and F. Hahne (2017). genefilter: genefilter: methods for filtering genes from high-throughput experiments. R package version 1.58.1
Paul Murrell (2009). Importing Vector Graphics: The grImport Package for R. Journal of Statistical Software, 30(4), 1-37. URL http://www.jstatsoft.org/v30/i04/
Baptiste Auguie (2017). gridExtra: Miscellaneous Functions for "Grid" Graphics. R package version 2.3. https://CRAN.R-project.org/package=gridExtra
R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. RL https://www.R-project.org/
https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg
Yu, G., 2020. ggplotify: Convert Plot to 'grob' or 'ggplot' Object. R package version 0.0.5.URLhttps://CRAN.R-project.org/package=ggplotify30
Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas
Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8
Guangchuang Yu (2020). ggplotify: Convert Plot to 'grob' or 'ggplot' Object. R package version 0.0.5. https://CRAN.R-project.org/package=ggplotify
Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Marques A et al. (2016). Oligodendrocyte heterogeneity in the mouse juvenile and adult central nervous system. Science 352(6291), 1326-1329. Amezquita R, Lun A, Becht E, Carey V, Carpp L, Geistlinger L, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks S (2020). "Orchestrating single-cell analysis with Bioconductor." Nature Methods, 17, 137–145. https://www.nature.com/articles/s41592-019-0654-x.

Examples

```
## In the following examples, the 2 toy data come from an RNA-seq analysis on development of 7
## chicken organs under 9 time points (Cardoso-Moreira et al. 2019). For conveninece, they are
## included in this package. The complete raw count data are downloaded using the R package
## ExpressionAtlas (Keays 2019) with the accession number "E-MTAB-6769". Toy data1 is used as
## a "data frame" input to exemplify data of simple samples/conditions, while toy data2 as
## "SummarizedExperiment" to illustrate data involving complex samples/conditions.
```

```
## Set up toy data.

# Access toy data1.
cnt.chk.simple <- system.file('extdata/shinyApp/data/count_chicken_simple.txt',
package='spatialHeatmap')
df.chk <- read.table(cnt.chk.simple, header=TRUE, row.names=1, sep='\t', check.names=FALSE)
# Columns follow the namig scheme "sample__condition", where "sample" and "condition" stands
# for organs and time points respectively.
df.chk[1:3, ]

# A column of gene annotation can be appended to the data frame, but is not required.
ann <- paste0('ann', seq_len(nrow(df.chk))); ann[1:3]
df.chk <- cbind(df.chk, ann=ann)
df.chk[1:3, ]

# Access toy data2.
cnt.chk <- system.file('extdata/shinyApp/data/count_chicken.txt', package='spatialHeatmap')
count.chk <- read.table(cnt.chk, header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing samples and conditions is required for toy data2. It should be made
# based on the experiment design, which is accessible through the accession number
# "E-MTAB-6769" in the R package ExpressionAtlas. An example targets file is included in this
# package and accessed below.
# Access the example targets file.
tar.chk <- system.file('extdata/shinyApp/data/target_chicken.txt', package='spatialHeatmap')
target.chk <- read.table(tar.chk, header=TRUE, row.names=1, sep='\t')
# Every column in toy data2 corresponds with a row in targets file.
target.chk[1:5, ]
# Store toy data2 in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)
# The "rowData" slot can store a data frame of gene annotation, but not required.
rowData(se.chk) <- DataFrame(ann=ann)

## As conventions, raw sequencing count data should be normalized, aggregated, and filtered to
## reduce noise.

# Normalize count data.
# The normalizing function "calcNormFactors" (McCarthy et al. 2012) with default settings
# is used.
df.nor.chk <- norm_data(data=df.chk, norm.fun='CNF', log2.trans=TRUE)
se.nor.chk <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)
# Aggregate count data.
# Aggregate "sample__condition" replicates in toy data1.
df.aggr.chk <- aggr_rep(data=df.nor.chk, aggr='mean')
df.aggr.chk[1:3, ]
# Aggregate "sample_condition" replicates in toy data2, where "sample" is "organism_part" and
# "condition" is "age".
se.aggr.chk <- aggr_rep(data=se.nor.chk, sam.factor='organism_part', con.factor='age',
aggr='mean')
assay(se.aggr.chk)[1:3, 1:3]
# Filter out genes with low counts and low variance. Genes with counts over 5 (log2 unit) in
```

```
# at least 1% samples (pOA), and coefficient of variance (CV) between 0.2 and 100 are retained.
# Filter toy data1.
df.fil.chk <- filter_data(data=df.aggr.chk, pOA=c(0.01, 5), CV=c(0.2, 100))
# Filter toy data2.
se.fil.chk <- filter_data(data=se.aggr.chk, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100))

## Spatial heatmaps.

# The target chicken aSVG is downloaded from the EBI aSVG repository
# (https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg) directly with
# function "return_feature". It is included in this package and accessed as below. Details on
# how this aSVG is selected are documented in function "return_feature".
svg.chk <- system.file("extdata/shinyApp/data", "gallus_gallus.svg",
package="spatialHeatmap")

# Reading the chicken aSVG file.
svg.chk <- read_svg(svg.path=svg.chk)

# Plot spatial heatmaps on gene "ENSGALG00000019846".
# Toy data1.
spatial_hm(svg=svg.chk, data=df.fil.chk, ID='ENSGALG00000019846', height=0.4,
legend.r=1.9, sub.title.size=7, ncol=3)
# Save spaital heatmaps as HTML and video files by assigning "out.dir" "~/test".

if (!dir.exists('~/test')) dir.create('~/test')
spatial_hm(svg=svg.chk, data=df.fil.chk, ID='ENSGALG00000019846', height=0.4,
legend.r=1.9, sub.title.size=7, ncol=3, out.dir='~/test')

# Toy data2.
spatial_hm(svg=svg.chk, data=se.fil.chk, ID='ENSGALG00000019846', legend.r=1.9,
legend.nrow=2, sub.title.size=7, ncol=3)
```

`SpatialEnrichment` *Identifying spatially enriched or depleted biomolecules*

Description

The spatial enrichment (SpEn) is designed to detect spatially enriched or depleted biomolecules (genes, proteins, etc) for chosen spatial features (cellular compartments, tissues, organs, *etc*). It compares each feature with all other reference features. The biomolecules significantly up- or down-regulated in one feature relative to reference features are denoted spatially enriched or depleted respectively. The underlying differential expression analysis methods include edgeR (Robinson et al, 2010), limma (Ritchie et al, 2015), and DESeq2 (Love et al, 2014). By querying a feature of interest from the enrichment results, the enriched or depleted biomolecules will be returned.
In addition, the SpEn is also able to identify biomolecules enriched or depleted in experiment vairables in a similar manner.

'sf_var()' subsets data according to given spatial features and variables.

'spatial_enrich()' detects enriched or depleted biomolecules for each given spatial feature.

'query_enrich()' queries enriched or depleted biomolecules in the enrichment results returned by `spatial_enrich` for a chosen spatial feature.

'ovl_enrich()' plots overlap of enrichment results across spatial features in form an upset plot, overlap matrix, or Venn diagram.

'graph_line()' plots expression values of chosen biomolecules in a line graph.

Usage

```
sf_var(
  data,
  feature,
  ft.sel = NULL,
  variable = NULL,
  var.sel = NULL,
  com.by = "ft"
)

spatial_enrich(
  data,
  method = c("edgeR"),
  norm = "TMM",
  m.array = FALSE,
  pairwise = FALSE,
  log2.fc = 1,
  p.adjust = "BH",
  fdr = 0.05,
  outliers = 0,
  aggr = "mean",
  log2.trans = TRUE,
  verbose = TRUE
)

query_enrich(res, query, other = FALSE, data.rep = FALSE)

ovl_enrich(
  res,
  type = "up",
  plot = "matrix",
  order.by = "freq",
  nintersects = 40,
  point.size = 3,
  line.size = 1,
  mb.ratio = c(0.6, 0.4),
  text.scale = 1.5,
  upset.arg = list(),
  show.plot = TRUE,
  venn.arg = list(),
```

```
  axis.agl = 45,
  font.size = 5,
  cols = c("lightcyan3", "darkorange")
)

graph_line(
  data,
  scale = "none",
  x.title = "Samples",
  y.title = "Assay values",
  linewidth = 1,
  text.size = 15,
  text.angle = 60,
  lgd.pos = "right",
  lgd.guide = guides(color = guide_legend(nrow = 1, byrow = TRUE, title = NULL))
)
```

Arguments

data	sf_var A SummarizedExperiment object. The colData slot is required to contain at least two columns of spatial features and experiment variables respectively. spatial_enrich A SummarizedExperiment object returned by sf_var. graph_line A data.frame, where rows are biomolecules and columns are spatial features.
feature	The column name in the colData slot of SummarizedExperiment that contains spatial features.
ft.sel	A vector of spatial features to choose.
variable	The column name in the colData slot of SummarizedExperiment that contains experiment variables.
var.sel	A vector of variables to choose.
com.by	One of ft, var, or ft.var. If ft, the enrichment is performed for each spatial feature and the variables are treated as replicates. If var the enrichment is performed for each variable and spatial features are treated as replicates. If ft.var, spatial features (tissue1, tissue2) and variables (var1, var2) are combined such as tissue1__var1, tissue1_var2, tissue2__var1, tissue2_var2. The enrichment is performed for each combination.
method	One of edgeR, limma, and DESeq2.
norm	The normalization method (TMM, RLE, upperquartile, none) in edgeR. The default is TMM. Details: https://www.rdocumentation.org/packages/edgeR/versions/3.14.0/topics/calcNormF
m.array	Logical. 'TRUE' and 'FALSE' indicate the input are microarray and count data respectively.
pairwise	Logical. If 'TRUE', pairwise comparisons will be performed starting dispersion estimation. If 'FALSE' (default), all samples are fitted into a GLM model together, then pairwise comparisons are performed through contrasts.
log2.fc	The log2-fold change cutoff. The default is 1.

p.adjust	The method (holm, hochberg, hommel, bonferroni, BH, BY, fdr, or none) for adjusting p values in multiple hypothesis testing. The default is BH.
fdr	The FDR cutoff. The default is 0.05.
outliers	The number of outliers allowed in the references. If there are too many references, there might be no enriched/depleted biomolecules in the query feature. To avoid this, set a certain number of outliers.
aggr	One of mean (default) or median. The method to aggregated replicates in the assay data.
log2.trans	Logical. If TRUE (default), the aggregated data (see aggr) is transformed to log2-scale and will be further used for plotting SHMs.
verbose	Logical. If 'TRUE' (default), intermmediate messages will be printed.
res	Enrichment results returned by spatial_enrich.
query	A spatial feature for query.
other	Logical (default is 'FALSE'). If 'TRUE' other genes that are neither enriched or depleted will also be returned.
data.rep	Logical. If 'TRUE' normalized data before aggregating replicates will be returned. If 'FALSE', normalized data after aggretating replicates will be returned.
type	One of up (default) or down, which refers to up- or down-regulated biomolecules.
plot	One of upset, matrix, or venn, corresponding to upset plot, overlap matrix, or Venn diagram respectively.
order.by	How the intersections in the matrix should be ordered by. Options include frequency (entered as "freq"), degree, or both in any order.
nintersects	Number of intersections to plot. If set to NA, all intersections will be plotted.
point.size	Size of points in matrix plot
line.size	The line thickness in overlap matrix.
mb.ratio	Ratio between matrix plot and main bar plot (Keep in terms of hundredths)
text.scale	Numeric, value to scale the text sizes, applies to all axis labels, tick labels, and numbers above bar plot. Can be a universal scale, or a vector containing individual scales in the following format: c(intersection size title, intersection size tick labels, set size title, set size tick labels, set names, numbers above bars)
upset.arg	A list of additional arguments passed to upset.
show.plot	Logical flag indicating whether the plot should be displayed. If false, simply returns the group count matrix.
venn.arg	A list of additional arguments passed to venn.
axis.agl	The angle of axis text in overlap matrix.
font.size	The font size of all text in overlap matrix.
cols	A vector of two colors indicating low and high values in the overlap matrix respectively. The default is c("lightcyan3", "darkorange").
scale	The method to scale the data. If none (default), no scaling. If row, each row is scaled independently. If all, all rows are scaled as a whole.

`x.title`, `y.title`
: The title of X-axis and Y-axis respectively.

`linewidth`
: The line width.

`text.size`
: The font size of all text.

`text.angle`
: The angle of axis text.

`lgd.pos`
: The position of legend. The default is `right`.

`lgd.guide`
: The `guides` function in `ggplot2` for customizing legends.

Value

'sf_var' A `SummarizedExperiment` object.

'spatial_enrich' A `list` object.

'query_enrich' A `SummarizedExperiment` object.

'ovl_enrich' An UpSet plot, overlap matrix plot, or Venn diagram.

'graph_line' A ggplot.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505–9 Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1 Robinson MD, McCarthy DJ and Smyth GK (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139-140 Ritchie, M.E., Phipson, B., Wu, D., Hu, Y., Law, C.W., Shi, W., and Smyth, G.K. (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Research 43(7), e47. Love, M.I., Huber, W., Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2 Genome Biology 15(12):550 (2014) Nils Gehlenborg (2019). UpSetR: A More Scalable Alternative to Venn and Euler Diagrams for Visualizing Intersecting Sets. R package version 1.4.0. https://CRAN.R-project.org/package=UpSetR H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016. Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/.

Examples

```
## In the following examples, the toy data come from an RNA-seq analysis on development of 7
## chicken organs under 9 time points (Cardoso-Moreira et al. 2019). For conveninece, it is
## included in this package. The complete raw count data are downloaded using the R package
## ExpressionAtlas (Keays 2019) with the accession number "E-MTAB-6769".
```

```
library(SummarizedExperiment)
# Access the count table.
cnt.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt', package='spatialHeatmap'), header=T
cnt.chk[1:3, 1:5]
# A targets file describing spatial features and conditions is required for toy data. It should be made
# based on the experiment design, which is accessible through the accession number
# "E-MTAB-6769" in the R package ExpressionAtlas. An example targets file is included in this
# package and accessed below.

# Access the example targets file.
tar.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt', package='spatialHeatmap'), header=
# Every column in count table corresponds with a row in targets file.
tar.chk[1:5, ]
# Store count data and targets file in "SummarizedExperiment".
se.chk <- SummarizedExperiment(assay=cnt.chk, colData=tar.chk)
# The "rowData" slot can store a data frame of gene metadata, but not required. Only the
# column named "metadata" will be recognized.
# Pseudo row metadata.
metadata <- paste0('meta', seq_len(nrow(cnt.chk))); metadata[1:3]
rowData(se.chk) <- DataFrame(metadata=metadata)

# Subset the count data by features (brain, heart, kidney) and variables (day10, day12).
# By setting com.by='ft', the subsequent spatial enrichment will be performed across
# features with the variables as replicates.
data.sub <- sf_var(data=se.chk, feature='organism_part', ft.sel=c('brain', 'kidney',
 'heart', 'liver'), variable='age', var.sel=c('day10', 'day35'), com.by='ft')

## As conventions, raw sequencing count data should be normalized and filtered to
## reduce noise. Since normalization will be performed in spatial enrichment, only filtering
## is required.

# Filter out genes with low counts and low variance. Genes with counts over 5 in
# at least 10% samples (pOA), and coefficient of variance (CV) between 3.5 and 100 are
# retained.
data.sub.fil <- filter_data(data=data.sub, sam.factor='organism_part', con.factor='age',
pOA=c(0.1, 5), CV=c(0.7, 100))

# Spatial enrichment for every spatial feature with 1 outlier allowed.
enr.res <- spatial_enrich(data.sub.fil, method=c('edgeR'), norm='TMM', log2.fc=1, fdr=0.05, outliers=1)
# Overlaps of enriched genes across features.
ovl_enrich(enr.res, type='up', plot='upset')
# Query the results for brain.
en.brain <- query_enrich(enr.res, 'brain')
rowData(en.brain)[1:3, c('type', 'total', 'method')]

# Read aSVG image into an "SVG" object.
svg.chk <- system.file("extdata/shinyApp/data", "gallus_gallus.svg",
package="spatialHeatmap")
svg.chk <- read_svg(svg.chk)
# Plot an enrichment SHM.
dat.enrich <- SPHM(svg=svg.chk, bulk=en.brain)
shm(data=dat.enrich, ID=rownames(en.brain)[1], legend.r=1, legend.nrow=7, sub.title.size=10, ncol=2, bar.width=0
# Line graph of gene expression profile.
```

```
graph_line(assay(en.brain[1, , drop=FALSE]), lgd.pos='bottom')
```

SPHM-class *The SPHM class*

Description

The SPHM class is designed to store numeric data and image objects for plotting spatial heatmaps.

Usage

```
SPHM(svg = NULL, bulk = NULL, cell = NULL, match = list(), output = list())
```

Arguments

svg	An SVG object containing one or multiple aSVG instances (see SVG and read_svg). In the aSVGs, spatial features (tissues, organs, etc) having counterparts with the same identifiers in the 'bulk' data will be colored accoording to expression profiles of chosen biomolecules (genes, proteins, etc).
bulk	The bulk data in form of numeric vector, data.frame, or SummarizedExperiment. See the 'data' argument of the function filter_data.
cell	The single-cell data in form of SingleCellExperiment. In the 'colData' slot, a column that stores cell group labels is required.
match	The list for re-matching in SHMs (only bulk data) or matching between single-cell and bulk data in co-visualization. **SHMs** A named list for rematching spatial features between numeric data (ftA, ftB) and aSVGs (ftC, ftD, ftE). In each slot, the slot name is a spatial feature from the data and the corresponding element is one or multiple spatial features from the aSVG. *E.g.* list(ftA = c('ftC', 'ftD'), ftB = c('ftE')). **Co-visualization plots** Mapping cells to tissues: a named list, where cell group labels from colData(sce. dimred)[, 'cell.group'] are the name slots and aSVG features are the corresponding list elements. Mapping tissues to cells: a named list, where tissues are the name slots and cells from colData(sce.dimred)[, 'cell.group'] are the corresponding list elements. Applicable when cell grouping methods are annodation labels, marker genes, clustering, or manual assignments.
output	A list of outputs, which is automatically generated.

Value

An SPHM object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

Examples

```
library(SummarizedExperiment)
# Import single-cell data.
sce.pa <- system.file("extdata/shinyApp/data", "cell_mouse_brain.rds",
package="spatialHeatmap")
sce <- readRDS(sce.pa)
# Pre-processing.
sce.dimred.quick <- process_cell_meta(sce)
colData(sce.dimred.quick)[1:3, 1:2]
sce.aggr.quick <- aggr_rep(sce.dimred.quick, assay.na='logcounts', sam.factor='label',
aggr='mean')
# Import the aSVG image.
svg.mus.brain.pa <- system.file("extdata/shinyApp/data", "mus_musculus.brain.svg",
package="spatialHeatmap")
svg.mus.brain <- read_svg(svg.mus.brain.pa)
# List for mapping single cells to bulk.
lis.match.quick <- list(hypothalamus=c('hypothalamus'), cortex.S1=c('cerebral.cortex', 'nose'))

# SPHM class for storing aSVG, bulk/sc data, and matching list.
dat.quick <- SPHM(svg=svg.mus.brain, bulk=sce.aggr.quick, cell=sce.dimred.quick,
match=lis.match.quick)

# Co-visualization plot.
# covis(data=dat.quick, ID=c('Apod'), dimred='PCA', cell.group='label',
# tar.cell=names(lis.match.quick), assay.na='logcounts', bar.width=0.11, dim.lgd.nrow=1,
# height=0.7, legend.r=1.5, legend.key.size=0.02, legend.text.size=12, legend.nrow=3)
```

SPHMMethods *Methods for S4 class* SPHM

Description

These are methods for getting or setting SPHM objects.

Usage

```
## S4 method for signature 'SPHM'
svg(x)

## S4 replacement method for signature 'SPHM'
svg(x) <- value

## S4 method for signature 'SPHM'
bulk(x)

## S4 replacement method for signature 'SPHM'
bulk(x) <- value
```

```
## S4 method for signature 'SPHM'
cell(x)

## S4 replacement method for signature 'SPHM'
cell(x) <- value

## S4 method for signature 'SPHM'
match(x)

## S4 replacement method for signature 'SPHM'
match(x) <- value

## S4 method for signature 'SPHM'
output(x)

## S4 replacement method for signature 'SPHM'
output(x) <- value

## S4 method for signature 'SPHM,ANY,ANY,ANY'
x[i]

## S4 replacement method for signature 'SPHM,ANY,ANY,ANY'
x[i] <- value

## S4 method for signature 'SPHM'
length(x)

## S4 method for signature 'SPHM'
name(x)
```

Arguments

x	An 'SPHM' object.
value	A value for replacement.
i	An integer specifying a slot of an SPHM object.

Value

An object of SVG, SummarizedExperiment, SingleCellExperiment, data.frame, numeric, or list.

Main methods

In the following code snippets, x is an SPHM object.

x[i], **svg(x), bulk(x), cell(x), match(x), output(x)** Subsetting a slot of x.

x[i <- value, svg(x) <- value, bulk(x) <- value, cell(x) <- value, match(x) <- value] Replacing a slot value in x.

`name(x)` All slot names in `x`.

`length(x)` Number of all slots in `x`.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

See Also

SPHM: creating SPHM objects.

Examples

```
# Import an aSVG image.
svg.hum.pa <- system.file("extdata/shinyApp/data", 'homo_sapiens.brain.svg',
package="spatialHeatmap")
svg.hum <- read_svg(svg.hum.pa)
# aSVG features.
feature.hum <- attribute(svg.hum)[[1]]
set.seed(20)
unique(feature.hum$feature)[1:10]

# Testing vector.
my_vec <- setNames(sample(1:100, 4), c('substantia.nigra', 'putamen', 'prefrontal.cortex',
'notMapped')); my_vec
# An SPHM object for storing bulk data and aSVG.
dat.quick <- SPHM(svg=svg.hum, bulk=my_vec)
# Getters and setters for SPHM objects.
svg(dat.quick); svg(dat.quick) <- svg.hum
dat.quick['bulk']; dat.quick['bulk'] <- my_vec
```

---

submatrix *Subsetting data matrix*

Description

Given one or multiple biomolecules (gene, protein, metabolite, *etc*) from a data matrix, this function subsets other biomolecules that have similar expression profiles with each of the given biomolecule independently. The subset data matrix of each biomolecule is combined in a row-wise manner and returned.

Usage

```
submatrix(
  data,
  assay.na = NULL,
  ID,
  p = 0.3,
```

```
  n = NULL,
  v = NULL,
  fun = "cor",
  cor.absolute = FALSE,
  arg.cor = list(method = "pearson"),
  arg.dist = list(method = "euclidean"),
  file = NULL
)
```

Arguments

data	A 'data.frame', 'SummarizedExperiment', or 'SingleCellExperiment' object, where the columns and rows of the data matrix are samples and biomolecules respectively. Since this function builds on co-expression analysis, samples should be at least 5, otherwise, the results are not reliable.
assay.na	Applicable when data is 'SummarizedExperiment' or 'SingleCellExperiment', where multiple assays could be stored. The name of assay to use.
ID	A vector of biomolecules of interest.
p	The proportion of top biomolecules with most similar expression profiles with a given biomolecule. Only biomolecules within this proportion are returned. It applies to each biomolecule independently and selected biomolecules of each given biomolecule are returned together.
n	An integer of top biomolecules with most similar expression profiles with a given biomolecule. Only biomolecules within this number are returned. It applies to each biomolecule independently and selected biomolecules of each given biomolecule are returned together.
v	A cutoff of correlation coefficient (CC, -1 to 1) or distance (>=0) for subsetting biomolecules sharing the most similar expression profiles with a given biomolecule. If fun='cor', only biomolecules with CC larger than v are returned. If fun='dist', only biomolecules with distance less than v are returned. It applies to each given biomolecule independently and selected biomolecules of each given biomolecule are returned together.
fun	The function to calculate similarity/distance measures, 'cor' (default) or 'dist', corresponding to cor or dist from the "stats" package respectively.
cor.absolute	Logical. If 'TRUE', absolute correlation coefficients (CCs) are used. Only applies to fun='cor'. Default is 'FALSE', meaning the CCs preserve the negative sign when subsetting biomolecules.
arg.cor	A list of arguments passed to cor in the "stats" package. Default is list(method="pearson").
arg.dist	A list of arguments passed to dist in the "stats" package. Default is list(method="euclidean").
file	The file name to save subset data matrix.

Value

The subset data matrix in form of 'data.frame', 'SummarizedExperiment', or 'SingleCellExperiment'.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Langfelder P and Horvath S, WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics 2008, 9:559 doi:10.1186/1471-2105-9-559 Peter Langfelder, Steve Horvath (2012). Fast R Functions for Robust Correlations and Hierarchical Clustering. Journal of Statistical Software, 46(11), 1-17. URL http://www.jstatsoft.org/v46/i11/ R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/ Peter Langfelder, Bin Zhang and with contributions from Steve Horvath (2016). dynamicTreeCut: Methods for Detection of Clusters in Hierarchical Clustering Dendrograms. R package version 1.63-1. https://CRAN.R-project.org/package=dynamicTreeCut Martin Morgan, Valerie Obenchain, Jim Hester and Hervé Pagès (2018). SummarizedExperiment: SummarizedExperiment container. R package version 1.10.1 Keays, Maria. 2019. ExpressionAtlas: Download Datasets from EMBL-EBI Expression Atlas Love, Michael I., Wolfgang Huber, and Simon Anders. 2014. "Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2." Genome Biology 15 (12): 550. doi:10.1186/s13059-014-0550-8 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. “Gene Expression Across Mammalian Organ Development.” Nature 571 (7766): 505–9

Examples

```
## The example data included in this package come from an RNA-seq analysis on
## development of 7 chicken organs under 9 time points (Cardoso-Moreira et al. 2019).
## The complete raw count data are downloaded using the R package ExpressionAtlas
## (Keays 2019) with the accession number "E-MTAB-6769".

# Access example count data.
count.chk <- read.table(system.file('extdata/shinyApp/data/count_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
count.chk[1:3, 1:5]

# A targets file describing spatial features and variables is made based on the
# experiment design.
target.chk <- read.table(system.file('extdata/shinyApp/data/target_chicken.txt',
package='spatialHeatmap'), header=TRUE, row.names=1, sep='\t')
# Every column in example data 2 corresponds with a row in the targets file.
target.chk[1:5, ]
# Store example data in "SummarizedExperiment".
library(SummarizedExperiment)
se.chk <- SummarizedExperiment(assay=count.chk, colData=target.chk)

# Normalize data.
se.chk.nor <- norm_data(data=se.chk, norm.fun='CNF', log2.trans=TRUE)

# Aggregate replicates of "spatialFeature_variable", where spatial features are organs
# and variables are ages.
se.chk.aggr <- aggr_rep(data=se.chk.nor, sam.factor='organism_part', con.factor='age',
aggr='mean')
```

```
assay(se.chk.aggr)[1:3, 1:3]

# Genes with experssion values >= 5 in at least 1% of all samples (pOA), and coefficient
# of variance (CV) between 0.2 and 100 are retained.
se.chk.fil <- filter_data(data=se.chk.aggr, sam.factor='organism_part', con.factor='age',
pOA=c(0.01, 5), CV=c(0.2, 100), file=NULL)

## Subset the data matrix for gene 'ENSGALG00000019846' and 'ENSGALG00000000112'.
se.sub.mat <- submatrix(data=se.chk.fil, ID=c('ENSGALG00000019846',
'ENSGALG00000000112'), p=0.1)

## Hierarchical clustering.
library(dendextend)
# Static matrix heatmap.
mhm.res <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=TRUE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
# Clusters containing "ENSGALG00000019846".
cut_dendro(mhm.res$rowDendrogram, h=15, 'ENSGALG00000019846')

# Interactive matrix heatmap.
 matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))


# In case the interactive heatmap is not automatically opened, run the following code snippet.
# It saves the heatmap as an HTML file that is assigned to the "file" argument.

mhm <- matrix_hm(ID=c('ENSGALG00000019846', 'ENSGALG00000000112'), data=se.sub.mat,
angleCol=80, angleRow=35, cexRow=0.8, cexCol=0.8, margin=c(8, 10), static=FALSE,
arg.lis1=list(offsetRow=0.01, offsetCol=0.01))
htmlwidgets::saveWidget(widget=mhm, file='mhm.html', selfcontained=FALSE)
browseURL('mhm.html')


## Adjacency matrix and module identification
adj.mod <- adj_mod(data=se.sub.mat)

# The adjacency is a measure of co-expression similarity between genes, where larger
# value denotes higher similarity.
adj.mod[['adj']][1:3, 1:3]

# The modules are identified at four sensitivity levels (ds=0, 1, 2, or 3). From 0 to 3,
# more modules are identified but module sizes are smaller. The 4 sets of module
# assignments are returned in a data frame, where column names are sensitivity levels.
# The numbers in each column are module labels, where "0" means genes not assigned to
# any module.
adj.mod[['mod']][1:3, ]

# Static network graph. Nodes are genes and edges are adjacencies between genes.
# The thicker edge denotes higher adjacency (co-expression similarity) while larger node
# indicates higher gene connectivity (sum of a gene's adjacencies with all its direct
```

```
# neighbors). The target gene is labeled by "_target".
network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, adj.min=0,
vertex.label.cex=1.5, vertex.cex=4, static=TRUE)

# Interactive network. The target gene ID is appended "_target".
 network(ID="ENSGALG00000019846", data=se.sub.mat, adj.mod=adj.mod, static=FALSE)
```

`SVG-class` *The SVG class for storing annotated SVG (aSVG) instances*

Description

The SVG class is designed to represent annotated SVG (aSVG) instances.

Usage

```
SVG(
  coordinate = list(),
  attribute = list(),
  dimension = list(),
  svg = list(),
  raster = list(),
  angle = list()
)
```

Arguments

`coordinate`	A named `list` of x-y coordinates parsed from one or multile aSVG files respectively. Coordinates are represented in three columns `x`, `y`, and `feature` in form of `data.frame` or `tbl`, corresponding to x, y coordinates, and spatial features (cellular compartments, tissues, organs, etc.) in aSVGs respectively. The `list` name slots refer to aSVG instances respectively, e.g. list(SVGInstance1=coordinate1, SVGInstance2=coordinate2).
`attribute`	A named `list` of attributes of coordinates in `coordinate`. Attributes are represented in at least four columns `feature`, `id`, `fill` and `stroke` in form of `data.frame` or `tbl`, corresponding to `feature` in `coordinate`, ids of `feature`, fill colors of `feature`, and line widths of `feature` respectively. `id` can be the same as `feature` or ontology ids. The `list` name slots refer to aSVG instances respectively and must match those in `coordinate`, e.g. list(SVGInstance1=attribute1, SVGInstance2=attribute2).
`dimension`	A named `list` of width/height parsed from one or multile aSVG files respectively, which is calculated from `coordinate` automatically. Each pair of width/height is stored in a named vector. The `list` name slots refer to aSVG instances respectively and must match those in `coordinate`, e.g. list(SVGInstance1=c(with=100, height=8), SVGInstance2=c(with=20, height=15)).

svg — A named list of one or multile aSVG instances, each in form of an 'raw' object returned by xml_serialize. The list name slots refer to aSVG files respectively and must match those in coordinate, e.g. list(SVGInstance1=svg1, SVGInstance2=svg2).

raster — A named list of directory path(s) of one or multile raster image files (jpg, png) respectively. This argument is relevant only when superimposing raster images with spatial heatmap plots that are created from aSVG images. The default is NULL for each aSVG instance. aSVG images are usually created by using these raster images as templates, otherwise spatial features between the two will not match. The list name slots refer to aSVG instances respectively and must match those in coordinate, e.g. list(SVGInstance1=raster.path1, SVGInstance2=raster.path2).

angle — Applicable in the case of spatially resolved single-cell data. A named list of one or multiple rotation degrees of the shapes with the identifier 'overlay' in one or muptiple aSVGs respectively. The list name slots refer to aSVG instances respectively and must be the same with coordinate, e.g. list(SVGInstance1=90, SVGInstance2=45).

Value

A SVG object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

Examples

```
# The first raste image used as a template to create an aSVG.
raster.pa1 <- system.file('extdata/shinyApp/data/maize_leaf_shm1.png',
package='spatialHeatmap')
# The first aSVG created with the first template.
svg.pa1 <- system.file('extdata/shinyApp/data/maize_leaf_shm1.svg',
package='spatialHeatmap')
# The second raster image used as a template to create an aSVG.
raster.pa2 <- system.file('extdata/shinyApp/data/maize_leaf_shm2.png',
package='spatialHeatmap')
# The second aSVG created with the second template.
svg.pa2 <- system.file('extdata/shinyApp/data/maize_leaf_shm2.svg',
package='spatialHeatmap')

# Parse these two aSVGs without association with raster images.
svgs <- read_svg(svg.path=c(svg.pa1, svg.pa2), raster.path=NULL)

# Parse these two aSVGs. The raster image paths are provide so as to
# be associated with respective aSVGs, which will be used when
# superimposing raster images with SHM plots.
svgs <- read_svg(svg.path=c(svg.pa1, svg.pa2), raster.path=c(raster.pa1, raster.pa2))
```

```
# Two aSVG instances are stored in a "SVG" object of "svgs".
names(svgs)
# Access content of "svgs".
svgs[1, ] # The first aSVG instance
svgs[, 'coordinate'][1]; coordinate(svgs)[1] # The coordinates of the first aSVG instance
# Combine two "SVG" objects.
x <- svgs[1, ]; y <- svgs[2, ]; cmb(x, y)
# Extract slots from "svgs" and create a new "SVG" object.
lis <- list(cordn=coordinate(svgs), attrb=attribute(svgs), svg=svg_obj(svgs), raster=raster_pa(svgs))
new.svgs <- SVG(coordinate=lis$cordn, attribute=lis$attrb, svg=lis$svg, raster=lis$raster)
# Change aSVG instance names.
names(new.svgs) <- c('aSVG1', 'aSVG2')
```

SVGMethods *Methods for S4 class* SVG

Description

These are methods for subsetting, getting, setting, or combining SVG objects.

Usage

```
## S4 method for signature 'SVG'
coordinate(x)

## S4 replacement method for signature 'SVG'
coordinate(x) <- value

## S4 method for signature 'SVG'
attribute(x)

## S4 replacement method for signature 'SVG'
attribute(x) <- value

## S4 method for signature 'SVG'
dimension(x)

## S4 replacement method for signature 'SVG'
dimension(x) <- value

## S4 method for signature 'SVG'
raster_pa(x)

## S4 replacement method for signature 'SVG'
raster_pa(x) <- value

## S4 method for signature 'SVG'
svg_obj(x)
```

```
## S4 replacement method for signature 'SVG'
svg_obj(x) <- value

## S4 method for signature 'SVG'
angle(x)

## S4 replacement method for signature 'SVG'
angle(x) <- value

## S4 method for signature 'SVG,ANY,ANY,ANY'
x[i, j]

## S4 replacement method for signature 'SVG,ANY,ANY,ANY'
x[i] <- value

## S4 method for signature 'SVG'
length(x)

## S4 method for signature 'SVG'
names(x)

## S4 replacement method for signature 'SVG'
names(x) <- value

## S4 method for signature 'SVG,SVG'
cmb(x, y)

## S4 method for signature 'SVG'
sub_sf(svg, show = NULL, hide = NULL)
```

Arguments

x, y	Two SVG objects.
value	A value for replacement.
i, j	Two integers specifying an aSVG instance and a slot of the same aSVG respectively.
svg	An SVG object.
show, hide	Two vectors of indexes in the attribute slot. aSVG features corresponding to these indexes will be shown or hidden in spatial heatmap plots respectively.

Value

An object of SVG, data.frame, or numeric.

Main methods

In the following code snippets, cordn is a SVG object.

`cordn[i]`, `cordn[i, ]` Subsetting the ith aSVG instance.

`cordn[i] <- cordn.new` Replacing the ith aSVG instance in `cordn` with a new `cordn` object `cordn.new`.

`cordn[, j]` Subsetting the jth slot of all aSVG instances.

`cordn[, 'coordinate']`, `coordinate(cordn)` Subsetting the `coordinate` slot that contains coordinates of all aSVG instances.

`length(cordn)` Number of all aSVG instances.

`names(cordn)`, `names(cordn)[1] <- 'newName'` Names of all aSVG instances, rename the first aSVG instance.

`cbm(cordn1, cordn2)` Combining two aSVG instances.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Wickham H, François R, Henry L, Müller K (2022). _dplyr: A Grammar of Data Manipulation_. R package version 1.0.9, <https:// CRAN.R-project.org/package=dplyr>

Wickham H, François R, Henry L, Müller K (2022). _dplyr: A Grammar of Data Manipulation_. R package version 1.0.9, <https://CRAN.R-project.org/package=dplyr>

See Also

SVG: creating SVG objects.

Examples

```
# Create the first aSVG instance.
svg.pa1 <- system.file('extdata/shinyApp/data/maize_leaf_shm1.svg',
package='spatialHeatmap')
svg1 <- read_svg(svg.path=c(svg.pa1)); names(svg1); length(svg1); slotNames(svg1)
# Create the second aSVG instance.
svg.pa2 <- system.file('extdata/shinyApp/data/maize_leaf_shm2.svg',
package='spatialHeatmap')
svg2 <- read_svg(svg.path=c(svg.pa2)); names(svg2); length(svg2)
# Combine these two instances.
svg3 <- cmb(svg1, svg2); names(svg3); length(svg3)
# The first aSVG instance
svg3[1]
# Coordinates of the first aSVG instance
svg3[, 'coordinate'][1]; coordinate(svg3)[1]
# Extract slots from "svg3" into a list and create a new "SVG" object.
lis <- list(cordn=coordinate(svg3), attrb=attribute(svg3), svg=svg_obj(svg3))
new.svgs <- SVG(coordinate=lis$cordn, attribute=lis$attrb, svg=lis$svg)
# Change aSVG instance names.
names(new.svgs) <- c('aSVG1', 'aSVG2'); names(new.svgs)
# Replace the second instance in "svg3".
svg3[2] <- new.svgs[2]
```

```
# Replace a slot content.
coordinate(svg3)[[1]] <- coordinate(new.svgs)[[1]]
```

`true_bulk` *Assign true bulk to cells in* `colData` *slot.*

Description

In co-clustering, assign true bulk to cells in `colData` slot.

Usage

```
true_bulk(sce, df.match)
```

Arguments

`sce`	A `SingleCellExperiment` of clustered single cell data.
`df.match`	The matching table between cells and true bulk.

Value

A `SingleCellExperiment` object.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Morgan M, Obenchain V, Hester J, Pagès H (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24. 0, https://bioconductor.org/packages/SummarizedExperiment.

Examples

```
# Matching table.
match.mus.brain.pa <- system.file("extdata/shinyApp/data", "match_mouse_brain_cocluster.txt", package="spatialHe
df.match.mus.brain <- read.table(match.mus.brain.pa, header=TRUE, row.names=1, sep='\t')
df.match.mus.brain

# Create random data matrix.
df.random <- matrix(rexp(30), nrow=5)
dimnames(df.random) <- list(paste0('gene', seq_len(nrow(df.random))), c('cere', 'cere', 'hipp', 'hipp', 'corti.su

library(SingleCellExperiment); library(S4Vectors)
cell.refined <- SingleCellExperiment(assays=list(logcounts=df.random), colData=DataFrame(cell=colnames(df.rando

#cell.refined <- true_bulk(cell.refined, df.match.mus.brain)
#colData(cell.refined)
```

```
# See detailed example in the "coclus_meta" function by running "?coclus_meta".
```

`update_feature` *Update aSVG Spatial Features*

Description

Successful spatial heatmap plotting requires the aSVG features of interest have matching samples (cells, tissues, *etc*) in the data. If this requirement is not fulfiled, either the sample identifiers in the data or the spatial feature identifiers in the aSVG should be changed. This function is designed to replace existing feature identifiers, stroke (outline) widths, and/or feature colors in aSVG files with user-provided entries.

Usage

```
update_feature(df.new, dir)
```

Arguments

`df.new`	The custom feature identifiers, stroke (outline) widths, and/or feature colors, should be included in the data frame returned by return_feature as independent columns, and the corresponding column names should be "featureNew", "strokeNew", and "colorNew" respectively in order to be recognized. To color the corresponding features, the identifiers in "featureNew" should be the same with matching sample identifiers. The numeric values in "strokeNew" would be the outline widths of corresponding features. The colors in "colorNew" would be the default colors for highlighting target features in the legend plot.
`dir`	The directory path where the aSVG files to update. It should be the same with `dir` in `return_feature`.

Value

Nothing is returned. The aSVG files of interest in `dir` are updated with provided attributes, and are ready to use in function `spatial_hm`.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Hadley Wickham, Jim Hester and Jeroen Ooms (2019). xml2: Parse XML. R package version 1.2.2. https://CRAN.R-project.org/package=xml2 Cardoso-Moreira, Margarida, Jean Halbert, Delphine Valloton, Britta Velten, Chunyan Chen, Yi Shao, Angélica Liechti, et al. 2019. "Gene Expression Across Mammalian Organ Development." Nature 571 (7766): 505-9 Gregory R. Warnes,

Ben Bolker, Lodewijk Bonebakker, Robert Gentleman, Wolfgang Huber, Andy Liaw, Thomas Lumley, Martin Maechler, Arni Magnusson, Steffen Moeller, Marc Schwartz and Bill Venables (2020). gplots: Various R Programming Tools for Plotting Data. R package version 3.0.3. https://CRAN.R-project.org/package=gplots

Examples

```
# The following shows how to download a chicken aSVG containing spatial features of 'brain'
# and 'heart' from the EBI aSVG repository directly
# (https://github.com/ebi-gene-expression-group/anatomogram/tree/master/src/svg). An empty
# directory is recommended so as to avoid overwriting existing SVG files with the same names.
# Here "~/test" is used.

# Remote aSVG repos.
data(aSVG.remote.repo)
tmp.dir <- normalizePath(tempdir(check=TRUE), winslash="/", mustWork=FALSE)
tmp.dir.ebi <- paste0(tmp.dir, '/ebi.zip')
tmp.dir.shm <- paste0(tmp.dir, '/shm.zip')

# Download the remote aSVG repos as zip files. According to Bioconductor's
# requirements, downloadings are not allowed inside functions, so the repos are
# downloaded before calling "return_feature".
download.file(aSVG.remote.repo$ebi, tmp.dir.ebi)
download.file(aSVG.remote.repo$shm, tmp.dir.shm)
remote <- list(tmp.dir.ebi, tmp.dir.shm)
# Make an empty directory "~/test" if not exist.
if (!dir.exists('~/test')) dir.create('~/test')
# Query the remote aSVG repos.
feature.df <- return_feature(feature=c('heart', 'brain'), species=c('gallus'), dir='~/test',
match.only=TRUE, remote=remote)
feature.df

# New features, stroke widths, colors.
ft.new <- c('BRAIN', 'HEART')
stroke.new <- c(0.05, 0.1)
col.new <- c('green', 'red')
# Include new features, stroke widths, colors to the feature data frame.
feature.df.new <- cbind(featureNew=ft.new, strokeNew=stroke.new, colorNew=col.new, feature.df)
feature.df.new

# Update features.
update_feature(df.new=feature.df.new, dir='~/test')
```

`write_svg` *Exporting each spatial heatmap to a separate SVG file*

Description

This function exports each spatial heatmap (associated with a specific gene and condition) as separate SVG files. In contrast to the original aSVG file, spatial features in the output SVG files are assigned heat colors.

Usage

```
write_svg(input, out.dir)
```

Arguments

input	The output returned by shm or covis.
out.dir	The directory path where the colored aSVG file will be saved.

Value

Nothing is returned.

Author(s)

Jianhai Zhang <jzhan067@ucr.edu>
Dr. Thomas Girke <thomas.girke@ucr.edu>

References

Hadley Wickham, Jim Hester and Jeroen Ooms (2019). xml2: Parse XML. R package version 1.2.2. https://CRAN.R-project.org/package=xml2

Examples

```
# Read the aSVG file.
svg.hum.pa <- system.file("extdata/shinyApp/data", 'homo_sapiens.brain.svg',
package="spatialHeatmap")
svg.hum <- read_svg(svg.hum.pa)
# Attributes of spatial features.
feature.hum <- attribute(svg.hum)[[1]]
set.seed(20) # To obtain reproducible results, a fixed seed is set.
# Spatial features.
unique(feature.hum$feature)[1:10]
# Create a random numeric vector.
my_vec <- setNames(sample(1:100, 4), c('substantia.nigra', 'putamen',
'prefrontal.cortex', 'notMapped'))
my_vec
# Plot spatial heatmaps with the imported aSVG and random vector.
dat.quick <- SPHM(svg=svg.hum, bulk=my_vec)
shm.res <- shm(data=dat.quick, ID='testing', ncol=1, sub.title.size=20, legend.nrow=3,
bar.width=0.1)
# Export each spatial heatmap (under a certain gene and condition) to a separate SVG
# file in a temporary directory.
write_svg(input=shm.res, out.dir=tempdir(check = TRUE))
```

Index

∗ **datasets**
aSVG.remote.repo, 14
∗ **spatial heatmap**
spatialHeatmap-package, 3
[,SPHM,ANY,ANY,ANY-method *(*SPHMMethods*)*, 119
[,SVG,ANY,ANY,ANY-method *(*SVGMethods*)*, 127
[<-,SPHM,ANY,ANY,ANY-method *(*SPHMMethods*)*, 119
[<-,SVG,ANY,ANY,ANY-method *(*SVGMethods*)*, 127

adj_mod, *6*, 7, *55, 56*, *94*
adjacency, *8*
aggr_rep, *6*, 11
angle *(*SVGMethods*)*, 127
angle,SVG-method *(*SVGMethods*)*, 127
angle<- *(*SVGMethods*)*, 127
angle<-,SVG-method *(*SVGMethods*)*, 127
aSVG.remote.repo, 14
attribute *(*SVGMethods*)*, 127
attribute,SVG-method *(*SVGMethods*)*, 127
attribute<- *(*SVGMethods*)*, 127
attribute<-,SVG-method *(*SVGMethods*)*, 127

BatchtoolsParam, *19*
bulk *(*SPHMMethods*)*, 119
bulk,SPHM-method *(*SPHMMethods*)*, 119
bulk<- *(*SPHMMethods*)*, 119
bulk<-,SPHM-method *(*SPHMMethods*)*, 119

calcNormFactors, *62, 63*
cell *(*SPHMMethods*)*, 119
cell,SPHM-method *(*SPHMMethods*)*, 119
cell<- *(*SPHMMethods*)*, 119
cell<-,SPHM-method *(*SPHMMethods*)*, 119
cell_group, 15
centerSizeFactors, *60*, *81*
chooseRnaHvgs.se, *81*, *88*
cluster_cell, 16
cluster_edge_betweenness, *17*, *22*
cluster_fast_greedy, *17*, *22*
cluster_leading_eigen, *17*, *22*
cluster_walktrap, *17*, *22*
cmb *(*SVGMethods*)*, 127
cmb,SVG,SVG-method *(*SVGMethods*)*, 127
coclus_opt, 18
cocluster, 20, *33*, *79*
com_factor, 26
coordinate *(*SVGMethods*)*, 127
coordinate,SVG-method *(*SVGMethods*)*, 127
coordinate<- *(*SVGMethods*)*, 127
coordinate<-,SVG-method *(*SVGMethods*)*, 127
cor, *122*
covis, 27, *133*
covis,SPHM-method *(*covis*)*, 27
cpm, *62*
custom_shiny, *6*, 36
cut_dendro, 40
cutreeHybrid, *9*
cv, *47*, *49*
cvt_id, 41

data_ref, 42
database, 43
desired_bulk_shiny *(*cocluster*)*, 20
dimension *(*SVGMethods*)*, 127
dimension,SVG-method *(*SVGMethods*)*, 127
dimension<- *(*SVGMethods*)*, 127
dimension<-,SVG-method *(*SVGMethods*)*, 127
dist, *122*

edit_tar, 46
estimateSizeFactors, *62, 63*

filter_asg *(*cocluster*)*, 20
filter_cell *(*cocluster*)*, 20
filter_data, *6*, *42, 43*, 47, *49*, *93, 94*, *118*

fread, *85*

ggplot, *52*
graph_line *(*SpatialEnrichment*)*, 112
guides, *116*

heatmap.2, *52*

length *(*SPHMMethods*)*, 119
length,SPHM-method *(*SPHMMethods*)*, 119
length,SVG-method *(*SVGMethods*)*, 127

make.names, *85*
makeKNNGraph, *17*, *22*
makeSNNGraph, *17*, *22*
match *(*SPHMMethods*)*, 119
match,SPHM-method *(*SPHMMethods*)*, 119
match<- *(*SPHMMethods*)*, 119
match<-,SPHM-method *(*SPHMMethods*)*, 119
matrix_hm, *6*, 51
MulticoreParam, *19*

name *(*SPHMMethods*)*, 119
name,SPHM-method *(*SPHMMethods*)*, 119
names *(*SVGMethods*)*, 127
names,SVG-method *(*SVGMethods*)*, 127
names<-,SVG-method *(*SVGMethods*)*, 127
network, *6*, *12*, *48*, 55, *62*
norm_cell, 59
norm_data, *6*, 61
norm_srsc, 64
normalizeRnaCounts.se, *60*, *81*

opt_bar, 66
opt_setting, 67
opt_violin, 68
optimal_k, 69
output *(*SPHMMethods*)*, 119
output,SPHM-method *(*SPHMMethods*)*, 119
output<- *(*SPHMMethods*)*, 119
output<-,SPHM-method *(*SPHMMethods*)*, 119
ovl_enrich *(*SpatialEnrichment*)*, 112

plot.default, *57*
plot_dim, 70
plot_kmeans, 72
plot_meta, 73
pOverA, *47*, *49*
process_cell_meta, 81

qc_cell, 83
query_enrich *(*SpatialEnrichment*)*, 112
quickRnaQc.se, *81*, *83*

raster_pa *(*SVGMethods*)*, 127
raster_pa,SVG-method *(*SVGMethods*)*, 127
raster_pa<- *(*SVGMethods*)*, 127
raster_pa<-,SVG-method *(*SVGMethods*)*, 127
read_cache, 84
read_fr, 85
read_hdf5, *43*
read_hdf5 *(*database*)*, 43
read_svg, 86, *105*, *118*
reduce_dim, 87
reduce_rep, 89
refine_asg *(*cocluster*)*, 20
return_feature, *6*, 90, *131*
rlog, *62*, *63*
runPca.se, *21*, *81*, *88*
runTsne.se, *81*, *88*
runUmap.se, *81*, *88*

save_cache, *84*, 92
saveRDS, *43*, *75*
sf_var *(*SpatialEnrichment*)*, 112
shiny_shm, *6*, *43*, *49*, 93
shm, *12*, *27*, *48*, *62*, 96, *133*
shm,SPHM-method *(*shm*)*, 96
spatial_enrich *(*SpatialEnrichment*)*, 112
spatial_hm, *6*, *90*, 103, *131*
spatial_hm,SVG-method *(*spatial_hm*)*, 103
SpatialEnrichment, 112
spatialHeatmap
*(*spatialHeatmap-package*)*, 3
spatialHeatmap-package, 3
SPHM, *29*, *75*, *98*, *105*, *119–121*
SPHM *(*SPHM-class*)*, 118
SPHM-class, 118
SPHMMethods, 119
Splicing, *34*, *80*, *101*, *109*
sub_sf *(*SVGMethods*)*, 127
sub_sf,SVG-method *(*SVGMethods*)*, 127
submatrix, *6–8*, *52*, *56*, *94*, 121
SVG, *87*, *105*, *118*, *127–129*
SVG *(*SVG-class*)*, 125
svg *(*SPHMMethods*)*, 119
svg,SPHM-method *(*SPHMMethods*)*, 119
SVG-class, 125
svg<- *(*SPHMMethods*)*, 119

svg<-,SPHM-method *(SPHMMethods)*, 119
svg_obj *(SVGMethods)*, 127
svg_obj,SVG-method *(SVGMethods)*, 127
svg_obj<- *(SVGMethods)*, 127
svg_obj<-,SVG-method *(SVGMethods)*, 127
SVGMethods, 127

TOMsimilarity, *8*
TOMsimilarityFromExpr, *8*
true_bulk, 130

update_feature, *6*, *90*, 131
upset, *115*

varianceStabilizingTransformation, *62*, *63*
venn, *115*

write_hdf5, *37*, *43*
write_hdf5 *(database)*, 43
write_svg, 132

xml_serialize, *126*