Package: CAESAR.Suite (via r-universe)

October 17, 2024

Type Package

Title CAESAR: a Cross-Technology and Cross-Resolution Framework for Spatial Omics Annotation

Version 0.1.0

Author Xiao Zhang [aut, cre], Wei Liu [aut], Jin Liu [aut]

Maintainer Xiao Zhang <zhangxiao1994@cuhk.edu.cn>

Description Biotechnology in spatial omics has advanced rapidly over the past few years, enhancing both throughput and resolution. However, existing annotation pipelines in spatial omics predominantly rely on clustering methods, lacking the flexibility to integrate extensive annotated information from single-cell RNA sequencing (scRNA-seq) due to discrepancies in spatial resolutions, species, or modalities. Here we introduce the CAESAR suite, an open-source software package that provides image-based spatial co-embedding of locations and genomic features. It uniquely transfers labels from scRNA-seq reference, enabling the annotation of spatial omics datasets across different technologies, resolutions, species, and modalities, based on the conserved relationship between signature genes and cells/locations at an appropriate level of granularity. Notably, CAESAR enriches location-level pathways, allowing for the detection of gradual biological pathway activation within spatially defined domain types. More details on the methods related to our paper currently under submission. A full reference to the paper will be provided in future versions once the paper is published.

License GPL (>= 2)

Depends R (>= 3.5.0)

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp (>= 1.0.10), Matrix, Seurat, DescTools, ProFAST, furrr, future, ggplot2, ggrepel, grDevices, irlba, pbapply, progress, scater, stats, ade4, methods

Repository CRAN

Encoding UTF-8

LazyData true

LazyDataCompression xz

VignetteBuilder knitr, rmarkdown

Suggests knitr, rmarkdown, cowplot, scales, tibble, dplyr, msigdbr

URL https://github.com/XiaoZhangryy/CAESAR.Suite

BugReports https://github.com/XiaoZhangryy/CAESAR.Suite/issues

RoxygenNote 7.3.2

NeedsCompilation yes

Date/Publication 2024-09-16 14:20:06 UTC

Contents

acc *Calculate Accuracy of Predicted Cell Types*

Description

This function calculates the accuracy of predicted cell types by comparing the predicted labels to the true labels. It allows for the exclusion of certain cell types from the reference when calculating accuracy.

Usage

```
acc(y.predict, y.truth, cts_notin_reference = NULL)
```

Arguments

y.predict	A character vector representing the predicted cell type labels.
y.truth	A character vector representing the true cell type labels.
cts_notin_reference	A character vector specifying the cell types that should be excluded from the accuracy calculation. If NULL, all cell types are included. Default is NULL.

Details

The function compares the predicted labels (‘y.predict‘) to the true labels (‘y.truth‘) and calculates the proportion of correct predictions. If ‘cts_notin_reference‘ is provided, cell types in this vector are excluded from the accuracy calculation.

Value

A numeric value representing the accuracy, i.e., the proportion of correctly predicted labels among the considered cell types.

Examples

```
# Example usage:
y.predict <- c("A", "B", "A", "C", "A", "B")
y.truth <- c("A", "B", "C", "C", "A", "A")
acc_value <- acc(y.predict, y.truth)
print(acc_value)
```

`add.gene.embedding` *Add Gene Embedding to Seurat Object*

Description

This function computes and adds gene embeddings to a Seurat object based on a provided adjacency matrix of spatial information and an existing cell embedding. It allows for the integration of gene-level information into the dimensional reduction of the Seurat object.

Usage

```
add.gene.embedding(seu, adjm, reduction.name = "caesar", assay = NULL)
```

Arguments

`seu`	A Seurat object. The Seurat object should contain pre-computed cell embeddings for dimensional reduction and expression data for genes.
`adjm`	A spatial adjacency matrix that represents relationships between cells or spots in the spatial transcriptomic data. This matrix is used to calculate the gene embeddings.
`reduction.name`	A character string specifying the name of the dimensional reduction method used for the cell embeddings (e.g., "ncfm", "caesar", etc.). The computed gene embeddings will be added to this slot. Default is "caesar".
`assay`	A character string specifying which assay to use from the Seurat object. If NULL, the function will use the default assay set in the Seurat object.

Value

The modified Seurat object with the computed gene embeddings added to the specified dimensional reduction (‘reduction.name‘).

Examples

```
data(toydata)

seu <- toydata$seu
pos <- toydata$pos

adjm <- ProFAST::AddAdj(as.matrix(pos), radius.upper = 200)
seu <- add.gene.embedding(
    seu = seu,
    adjm = adjm,
    reduction.name = "caesar",
    assay = "RNA"
)
print(seu)
```

annotation_mat *Annotate Cells Using Distance Matrix and Marker Frequencies*

Description

This function annotates cells based on a cell-gene distance matrix and marker gene frequencies. It computes the average distances, optionally calculates confidence levels for the predictions, and computes cell mixing proportions.

Usage

```
annotation_mat(
  distce,
  marker.freq,
  gene.use = NULL,
  cal.confidence = TRUE,
  cal.proportions = TRUE,
  parallel = TRUE,
  ncores = 10,
  n_fake = 1001,
  seed = 1,
  threshold = 0.95,
  unassign = "unassigned"
)
```

Arguments

distce	A matrix of distances between spots and genes. Rows represent genes, and columns represent cells. Generally, it is a list of the output of function ProFAST::pdistance with CAESAR co-embedding as input.
marker.freq	A matrix where rows represent cell types, and columns represent marker genes. The values in the matrix represent the frequency or weight of each marker gene for each cell type. Generally, it is a list of the output of function markerList2mat.
gene.use	A character vector specifying which genes to use for the annotation. If 'NULL', all genes in 'distce' will be used. Default is 'NULL'.
cal.confidence	Logical, indicating whether to calculate the confidence of the predictions. Default is 'TRUE'.
cal.proportions	Logical, indicating whether to calculate the mixing proportions of cell types for each spot. Default is 'TRUE'.
parallel	Logical, indicating whether to run the confidence calculation in parallel. Default is 'TRUE'.
ncores	The number of cores to use for parallel computation. Default is 10.
n_fake	The number of fake (randomized) distance matrices to simulate for confidence calculation. Default is 1001.

seed	The random seed for reproducibility. Default is 1.
threshold	A numeric value specifying the confidence threshold below which a cell is labeled as 'unassigned'. Default is 0.95.
unassign	A character string representing the label to assign to cells below the confidence threshold. Default is '"unassigned"'.

Value

A list with the following components:

ave.dist	A matrix of average distances between each cell and each cell type.
confidence	A numeric vector of confidence values for each cell (if 'cal.confidence = TRUE').
pred	A character vector of predicted cell types for each cell.
pred_unassign	A character vector of predicted cell types with cells below the confidence threshold labeled as 'unassigned' (if 'cal.confidence = TRUE').
cell_mixing_proportions	A matrix of mixing proportions for each spot across the different cell types (if 'cal.proportions = TRUE').

See Also

marker.select for select markers. find.sig.genes for signature gene list. markerList2mat for marker frequency matrix. pdistance for obtain cell-gene distance matrix using co-embedding.

Examples

```
data(toydata)

seu <- toydata$seu
markers <- toydata$markers

seu <- ProFAST::pdistance(seu, reduction = "caesar")
distce <- Seurat::GetAssayData(object = seu, slot = "data", assay = "distce")

marker.freq <- markerList2mat(list(markers))

anno_res <- annotation_mat(distce, marker.freq, cal.confidence = FALSE, cal.proportions = FALSE)
str(anno_res)
```

---

auc *Calculate Area Under the Curve (AUC) for Pathway Scores*

Description

This function calculates the area under the curve (AUC) for pathway scores with respect to a specific cell type. It uses the AUC to evaluate the performance of the pathway scores in distinguishing the target cell type from others.

Usage

```
auc(celltype, pathway.scores, return.mean = TRUE, seed = 1)
```

Arguments

celltype	A factor or character vector representing the cell type labels for each cell.
pathway.scores	A matrix of pathway scores where rows represent cells and columns represent different pathways.
return.mean	Logical, indicating whether to return the weighted mean AUC across all cell types. If FALSE, returns the AUC for each pathway. Default is TRUE.
seed	An integer specifying the random seed for reproducibility. Default is 1.

Value

If return.mean = TRUE, a numeric vector containing the weighted mean AUC. If return.mean = FALSE, a numeric matrix of AUCs where rows represent pathways and columns represent cell types.

See Also

AUC for the AUC calculation.

Examples

```
# Example usage:
celltype <- factor(rep(letters[1:5], each = 20))
pathway.scores <- matrix(runif(1000), nrow = 100, ncol = 10)
colnames(pathway.scores) <- letters[1:10]
auc_values <- auc(celltype, pathway.scores, return.mean = TRUE)
print(auc_values)
```

CAESAR.annotation *Perform Cell Annotation Using CAESAR with Confidence and Proportion Calculation*

Description

This function annotates cells in a Seurat object using marker gene frequencies and a distance matrix. It calculates average distances between cells and cell types, confidence scores, and mixing proportions. Optionally, it can add the annotations and related metrics to the Seurat object metadata.

Usage

```
CAESAR.annotation(
  seu,
  marker.freq,
  reduction.name = "caesar",
  assay.dist = "distce",
  gene.use = NULL,
  cal.confidence = TRUE,
  cal.proportions = TRUE,
  parallel = TRUE,
  ncores = 10,
  n_fake = 1001,
  seed = 1,
  threshold = 0.95,
  unassign = "unassigned",
  add.to.meta = FALSE
)
```

Arguments

seu	A Seurat object containing cell expression data.
marker.freq	A matrix where rows represent cell types and columns represent marker genes. The values in the matrix represent the frequency or weight of each marker gene for each cell type. Generally, it is a list of the output of function markerList2mat.
reduction.name	A character string specifying the name of the dimensional reduction to use when calculating distances. Default is "caesar".
assay.dist	A character string specifying the name of the assay to store the distance matrix. If not present in the Seurat object, the function will calculate the distances using ProFAST::pdistance. Default is "distce".
gene.use	A character vector specifying which genes to use for the annotation. If NULL, all genes in the distance matrix will be used. Default is NULL.
cal.confidence	Logical, indicating whether to calculate the confidence of the predictions. Default is TRUE.
cal.proportions	Logical, indicating whether to calculate the mixing proportions of cell types for each cell. Default is TRUE.
parallel	Logical, indicating whether to run the confidence calculation in parallel. Default is TRUE.
ncores	The number of cores to use for parallel computation. Default is 10.
n_fake	The number of fake (randomized) distance matrices to simulate for confidence calculation. Default is 1001.
seed	The random seed for reproducibility. Default is 1.
threshold	A numeric value specifying the confidence threshold below which a cell is labeled as unassigned. Default is 0.95.

unassign	A character string representing the label to assign to cells below the confidence threshold. Default is "unassigned".
add.to.meta	Logical, indicating whether to return the annotation results directly or add them to the Seurat object metadata. If TRUE, the function will return the results directly. Default is FALSE.

Value

If add.to.meta = FALSE, the Seurat object with the added metadata for predicted cell types (CAESAR), predictions with unassigned (CAESARunasg), confidence scores (CAESARconf), average distances, and mixing proportions. If add.to.meta = TRUE, a list containing the above annotation results is returned.

See Also

marker.select for select markers. find.sig.genes for signature gene list. markerList2mat for marker frequency matrix. pdistance for obtain cell-gene distance matrix using co-embedding. annotation_mat for annotation procedure.

Examples

```
data(toydata)

seu <- toydata$seu
markers <- toydata$markers

marker.freq <- markerList2mat(list(markers))
anno_res <- CAESAR.annotation(seu, marker.freq, cal.confidence = FALSE, cal.proportions = FALSE)
str(anno_res)
```

CAESAR.coembedding *Compute Co-embedding Using CAESAR*

Description

This function performs co-embedding of both cells and genes using the CAESAR method. It integrates spatial transcriptomics data from a Seurat object ('seu') with a spatial adjacency matrix to compute the low-dimensional co-embedding.

Usage

```
CAESAR.coembedding(
  seu,
  pos,
  reduction.name = "caesar",
  q = 50,
  radius.upper = 400,
  ...
)
```

Arguments

seu	A Seurat object containing spatial transcriptomics data.
pos	A matrix of spatial coordinates for the spots (e.g., spatial positions of cells or pixels in the image). The row names of 'pos' should match the column names of 'seu'.
reduction.name	A character string specifying the name of the dimensional reduction method to store in the Seurat object. Default is "caesar".
q	An integer specifying the number of dimensions for the reduced co-embeddings. Default is 50.
radius.upper	A numeric value specifying the upper limit of the search radius for the spatial adjacency matrix. Default is 400.
...	Additional arguments passed to 'cellembedding_image_seurat'.

Value

The modified Seurat object with the computed cell and gene embeddings stored in the specified reduction slot.

See Also

cellembedding_seurat for computing cell embeddings. add.gene.embedding for adding gene embeddings to a Seurat object.

Examples

```
data(toydata)

seu <- toydata$seu
pos <- toydata$pos

seu <- CAESAR.coembedding(
    seu = seu,
    pos = pos
)
print(seu)
```

CAESAR.coembedding.image

Compute Co-embedding with Image Information Using CAESAR

Description

This function performs co-embedding of both cells and genes using the CAESAR method. It integrates spatial transcriptomics data from a Seurat object ('seu') with image features ('feature_img') and a spatial adjacency matrix to compute the low-dimensional co-embedding.

Usage

```
CAESAR.coembedding.image(
  seu,
  feature_img,
  pos,
  reduction.name = "caesar",
  q = 50,
  lower.med = 3.5,
  upper.med = 5.5,
  radius.upper = 400,
  q.image = 10,
  weighted = FALSE,
  approx_Phi = TRUE,
  seed = 1,
  ...
)
```

Arguments

seu	A Seurat object containing spatial transcriptomics data.
feature_img	A matrix representing features extracted from a histology image using a Visual Transformer. Rows correspond to spots and columns represent image features. The row names of 'feature_img' should match the column names of 'seu'.
pos	A matrix of spatial coordinates for the spots (e.g., spatial positions of cells or pixels in the image). The row names of 'pos' should match the column names of 'seu'.
reduction.name	A character string specifying the name of the dimensional reduction method to store in the Seurat object. Default is "caesar".
q	An integer specifying the number of dimensions for the reduced co-embeddings. Default is 50.
lower.med	A numeric value specifying the lower bound for the desired median number of neighbors in the spatial adjacency matrix. Default is 3.5.
upper.med	A numeric value specifying the upper bound for the desired median number of neighbors in the spatial adjacency matrix. Default is 5.5.
radius.upper	A numeric value specifying the upper limit of the search radius for the spatial adjacency matrix. Default is 400.
q.image	An integer specifying the number of dimensions for the reduced image embeddings. Default is 10.
weighted	Logical, indicating whether to apply weighted PCA on the image features. Default is FALSE.
approx_Phi	Logical, indicating whether to use an approximate method for Phi matrix estimation. Default is TRUE.
seed	An integer used to set the random seed for reproducibility. Default is 1.
...	Additional arguments passed to 'cellembedding_image_seurat'.

Value

The modified Seurat object with the computed cell and gene embeddings stored in the specified reduction slot.

See Also

`cellembedding_image_seurat` for computing cell embeddings. `add.gene.embedding` for adding gene embeddings to a Seurat object.

Examples

```
data(toydata)

seu <- toydata$seu
pos <- toydata$pos
imgf <- toydata$imgf

seu <- CAESAR.coembedding.image(seu, imgf, pos)
print(seu)
```

CAESAR.CTDEP *Test Cell Type Differentially Enriched Pathways*

Description

This function tests whether specific pathways are differentially enriched in particular cell types with cell types stored in a Seurat object. The function applies the Wilcoxon rank-sum test to compare the pathway scores between cells of a given cell type and all other cells. It supports parallel computation to speed up the testing process.

Usage

```
CAESAR.CTDEP(
  seu,
  pathway_scores,
  ident = NULL,
  cts = NULL,
  parallel = TRUE,
  ncores = 10,
  seed = 1
)
```

Arguments

`seu`	A Seurat object containing the single-cell RNA-seq data.
`pathway_scores`	A matrix of pathway scores where rows represent cells and columns represent different pathways.

ident	A character string specifying the column name in the Seurat object's metadata to use as the cell type labels. If NULL, the default identities (Idents(seu)) will be used. Default is NULL.
cts	A character vector specifying the cell types to test. If NULL, the function will test all unique cell types present in ident or Idents(seu). Default is NULL.
parallel	Logical, indicating whether to run the computation in parallel. Default is TRUE.
ncores	An integer specifying the number of cores to use for parallel computation. Default is 10.
seed	An integer specifying the random seed for reproducibility in parallel computation. Default is 1.

Value

A matrix of p-values where rows represent the pathways and columns represent the tested cell types.

See Also

wilcox.test for the Wilcoxon rank-sum test. CAESAR.enrich.score for spot level pathway enrich scores.

Examples

```
data(toydata)

seu <- toydata$seu
pathway_list <- toydata$pathway_list
cts <- levels(seu)[1:2]

enrich.score <- CAESAR.enrich.score(seu, pathway_list)
dep.pvals <- CAESAR.CTDEP(seu, enrich.score, cts = cts)
print(dep.pvals)
```

CAESAR.enrich.pathway *Test whether pathways are enriched*

Description

This function tests whether pathways are enriched. Specifically, for a pathway (gene set), the function will assess whether these genes are clustered in the embedding space. For more details, see Bai and Chu (2023).

Usage

```
CAESAR.enrich.pathway(
  seu,
  pathway.list,
  reduction = "caesar",
  pathway.cutoff = 3,
  test.type = list("ori", "gen", "wei", "max"),
  k = 5,
  wei.fun = c("weiMax", "weiGeo", "weiArith"),
  perm.num = 0,
  progress_bar = TRUE,
  ncores = 10,
  eta = 1e-04,
  genes.use = NULL,
  parallel = TRUE
)
```

Arguments

seu	A Seurat object containing co-embedding.
pathway.list	A list of pathways, where each pathway is characterized as a vector of gene sets.
reduction	The embedding used when a Seurat object is provided. Default is "caesar".
pathway.cutoff	The minimal number of genes required in a pathway. Pathways with fewer genes than this threshold will not be considered in the enrichment test.
test.type	Type of graph-based test. This must be a list containing elements chosen from "ori", "gen", "wei", and "max", with default ‘list("ori", "gen", "wei", "max")‘. "ori" refers to robust original edge-count test, "gen" refers to robust generalized edge-count test, "wei" refers to robust weighted edge-count test, and "max" refers to robust max-type edge-count tests. For more details, see Bai and Chu (2023).
k	The parameter for k-minimum spanning tree. Default is 5.
wei.fun	The weighted function used in the enrichment test. Default is "weiMax", which returns the inverse of the max node degree of an edge. "weiGeo" returns the inverse of the geometric average of the node degrees, and "weiArith" returns the inverse of the arithmetic average of the node degrees.
perm.num	The number of permutations used to calculate the p-value (default is 0). Use 0 for getting only the approximate p-value based on asymptotic theory. Setting perm.num (e.g., perm.num = 1000) allows permutation-based p-value calculation, though this may be time-consuming.
progress_bar	Logical, TRUE or FALSE, indicating whether a progress bar should be printed during permutations. Default is TRUE.
ncores	The number of cores to use for parallel computing. Default is 10.
eta	A small positive number to ensure matrix inversion stability. Default is 1e-4.
genes.use	A vector of genes representing a gene set. All pathways will be tested for enrichment after intersecting with this gene set.

`parallel` Logical, indicating whether to use parallel computing to speed up computation. Default is TRUE.

Value

A data.frame containing the results of the test with the following columns:

- asy.ori.statistic - test statistic of robust original edge-count test.
- asy.ori.pval - asymptotic theory based p value of robust original edge-count test.
- asy.ori.pval.adj - the adjusted asymptotic theory based p value of robust original edge-count test.
- perm.ori.pval - permutation-based p value of robust original edge-count test, appear when permutation-based p-value is calculated.
- perm.ori.pval.adj - the adjusted permutation-based p value of robust original edge-count test, appear when permutation-based p-value is calculated.
- asy.gen.statistic - test statistic of robust generalized edge-count test.
- asy.gen.pval - asymptotic theory based p value of robust generalized edge-count test.
- asy.gen.pval.adj - the adjusted asymptotic theory based p value of robust generalized edge-count test.
- perm.gen.pval - permutation-based p value of robust generalized edge-count test, appear when permutation-based p-value is calculated.
- perm.gen.pval.adj - the adjusted permutation-based p value of robust generalized edge-count test, appear when permutation-based p-value is calculated.
- asy.wei.statistic - test statistic of robust weighted edge-count test.
- asy.wei.pval - asymptotic theory based p value of robust weighted edge-count test.
- asy.wei.pval.adj - the adjusted asymptotic theory based p value of robust weighted edge-count test.
- perm.wei.pval - permutation-based p value of robust weighted edge-count test, appear when permutation-based p-value is calculated.
- perm.wei.pval.adj - the adjusted permutation-based p value of robust weighted edge-count test, appear when permutation-based p-value is calculated.
- asy.max.statistic - test statistic of robust max-type edge-count tests.
- asy.max.pval - asymptotic theory based p value of robust max-type edge-count tests.
- asy.max.pval.adj - the adjusted asymptotic theory based p value of robust max-type edge-count tests.
- perm.max.pval - permutation-based p value of robust max-type edge-count tests, appear when permutation-based p-value is calculated.
- perm.max.pval.adj - the adjusted permutation-based p value of robust max-type edge-count tests, appear when permutation-based p-value is calculated.

References

Bai, Y., & Chu, L. (2023). A Robust Framework for Graph-based Two-Sample Tests Using Weights. arXiv preprint arXiv:2307.12325.

Examples

```
data(toydata)

seu <- toydata$seu
pathway_list <- toydata$pathway_list

CAESAR.enrich.pathway(seu, pathway_list)
```

CAESAR.enrich.score *Calculate Spot Level Enrichment Scores for Pathways Using CAESAR*

Description

This function calculates spot level enrichment scores for a list of pathways based on a cell-gene distance matrix in a Seurat object. The function uses a permutation-based approach to determine the significance of the enrichment scores.

Usage

```
CAESAR.enrich.score(
  seu,
  pathwaylist,
  assay.dist = "distce",
  reduction.name = "caesar",
  gene.use = NULL,
  n_fake = 1001,
  seed = 1
)
```

Arguments

seu	A Seurat object containing the gene expression data.
pathwaylist	A list of pathways, where each pathway is represented by a vector of genes.
assay.dist	A character string specifying the assay that contains the distance matrix. Default is "distce".
reduction.name	A character string specifying the reduction method to use if the distance matrix needs to be computed. Default is "caesar".
gene.use	A character vector specifying which genes to use in the analysis. If NULL, all genes in the distance matrix will be used. Default is NULL.
n_fake	An integer specifying the number of random permutations to generate for significance testing. Default is 1001.
seed	An integer specifying the random seed for reproducibility. Default is 1.

Value

A matrix of enrichment scores with cells as rows and pathways as columns.

Examples

```
data(toydata)

seu <- toydata$seu
pathway_list <- toydata$pathway_list

enrich.score <- CAESAR.enrich.score(seu, pathway_list)
head(enrich.score)
```

CAESAR.RUV *Perform Batch Correction and Integration with CAESAR Using Housekeeping Genes*

Description

This function performs batch correction and integration of multiple Seurat objects using housekeeping genes and distance matrices. It supports human and mouse data, and can optionally use custom housekeeping genes provided by the user.

Usage

```
CAESAR.RUV(
  seuList,
  distList,
  verbose = FALSE,
  species = "human",
  custom_housekeep = NULL
)
```

Arguments

seuList	A list of Seurat objects to be integrated.
distList	A list of distance matrices corresponding to each Seurat object in 'seuList'.
verbose	Logical, indicating whether to display progress messages. Default is FALSE.
species	A character string specifying the species, either "human" or "mouse". Default is "human".
custom_housekeep	A character vector of custom housekeeping genes. If NULL, default housekeeping genes for the species are used. Default is NULL.

Value

A Seurat object that contains the integrated and batch-corrected data in a new assay called "CAESAR".

Examples

```
data(toydata)

seu <- toydata$seu
markers <- toydata$markers

seu <- ProFAST::pdistance(seu, reduction = "caesar")

marker.freq <- markerList2mat(list(markers))
anno_res <- CAESAR.annotation(seu, marker.freq, cal.confidence = FALSE, cal.proportions = FALSE)

seuList <- list(seu, seu)
distList <- list(anno_res$ave.dist, anno_res$ave.dist)
seuInt <- CAESAR.RUV(seuList, distList, species = "human", verbose = TRUE)
```

Cauchy.Combination *Combine p-values Using the Cauchy Combination Method*

Description

This function combines multiple p-values using the Cauchy combination method. The method is particularly useful for combining dependent p-values and is known for being robust and powerful, especially in the context of small p-values. For details, see Liu and Xie (2020).

Usage

```
Cauchy.Combination(pvals, weight = NULL)
```

Arguments

pvals	A numeric vector of p-values to be combined. The p-values should be between 0 and 1.
weight	A numeric vector of weights corresponding to the p-values. If NULL, equal weights are assigned to all p-values. Default is NULL.

Value

A single combined p-value.

References

Liu, Y., & Xie, J. (2020). Cauchy combination test: a powerful test with analytic p-value calculation under arbitrary dependency structures. Journal of the American Statistical Association, 115(529), 393-402.

Examples

```
# Example usage:
pvals <- c(0.01, 0.03, 0.05)
combined_pval <- Cauchy.Combination(pvals)
print(combined_pval)
```

`cellembedding_image_matrix` *Compute Spatial-Aware Cell Embeddings with Image Information*

Description

This function computes low-dimensional cell embeddings from a gene-by-cell matrix. The method initializes cell embeddings using approximate PCA and refines them through a linear factor model nested a intrinsical conditional autoregressive model.

Usage

```
cellembedding_image_matrix(
  X,
  adjm,
  q = 50,
  reduction.name = "caesar",
  maxIter = 30,
  epsELBO = 1e-06,
  approx_Phi = FALSE,
  verbose = TRUE,
  Phi_diag = TRUE,
  seed = 1
)
```

Arguments

`X`	A gene-by-cell matrix (e.g., the 'data' slot from a Seurat object) that serves as the input data for dimensional reduction.
`adjm`	A spatial adjacency matrix representing relationships between cells or spots.
`q`	An integer specifying the number of dimensions for the reduced embeddings. Default is 50.
`reduction.name`	A character string specifying the name of the dimensional reduction method. Default is 'caesar'.
`maxIter`	Maximum number of iterations for the optimization algorithm. Default is 30.
`epsELBO`	A small number specifying the convergence threshold for the optimization algorithm. Default is 1e-6.

approx_Phi	Logical, indicating whether to use the approximate method for Phi matrix estimation. Default is FALSE.
verbose	Logical, indicating whether to print progress messages. Default is TRUE.
Phi_diag	Logical, indicating whether to constrain the Phi matrix to be diagonal. Default is TRUE.
seed	An integer used to set the random seed for reproducibility. Default is 1.

Value

A matrix containing the computed cell embeddings. The number of rows corresponds to the number of cells, and the number of columns corresponds to the specified number of dimensions (‘q‘).

---

cellembedding_image_seurat

Compute Spatial-Aware Cell Embeddings with Image Information

Description

This function computes low-dimensional cell embeddings from a seyrat object. The method initializes cell embeddings using approximate PCA and refines them through a linear factor model nested a intrinsical conditional autoregressive model.

Usage

```
cellembedding_image_seurat(
  seu,
  adjm,
  assay = NULL,
  slot = "data",
  q = 10,
  approx_Phi = FALSE,
  reduction.name = "caesar",
  var.features = NULL,
  ...
)
```

Arguments

seu	A Seurat object containing gene expression data. The object should have variable features identified prior to running this function.
adjm	A spatial adjacency matrix representing relationships between cells or spots.
assay	A character string specifying which assay to use from the Seurat object. If NULL, the function will use the default assay set in the Seurat object.
slot	The data slot to use for feature extraction (e.g., "data", "scale.data"). Default is "data".

q	An integer specifying the number of dimensions for the reduced embeddings. Default is 10.
approx_Phi	Logical, indicating whether to use an approximate method for estimating the Phi matrix. Default is FALSE.
reduction.name	A character string specifying the name for the dimensional reduction result. Default is "caesar".
var.features	A vector of variable features (genes) to use for the analysis. If NULL, the function will automatically use the variable features stored in the Seurat object.
...	Additional arguments passed to 'cellembedding_image_matrix'.

Value

The modified Seurat object with the cell embedding results stored in the specified dimensional reduction slot.

See Also

cellembedding_image_matrix for additional arguments used to compute cell embeddings.

---

cellembedding_matrix *Compute Spatial-Aware Cell Embeddings*

Description

This function computes low-dimensional cell embeddings from a gene-by-cell matrix. The method initializes cell embeddings using approximate PCA and refines them through a linear factor model nested a intrinsical conditional autoregressive model.

Usage

```
cellembedding_matrix(X, adjm, q = 50, reduction.name = "caesar", ...)
```

Arguments

X	A gene-by-cell matrix (e.g., the 'data' slot from a Seurat object) that serves as the input data for dimensional reduction.
adjm	A spatial adjacency matrix representing the relationships between cells or spots in spatial transcriptomic data.
q	An integer specifying the number of dimensions to reduce to. Default is 50.
reduction.name	A character string specifying the name of the dimensional reduction method. Default is 'caesar'.
...	Additional parameters passed to 'ProFAST::FAST_run'.

Value

A matrix containing the computed cell embeddings. The number of rows corresponds to the number of cells, and the number of columns corresponds to the specified number of dimensions (‘q‘).

See Also

FAST_run for the main FAST dimensionality reduction algorithm.

Examples

```
data(toydata)

seu <- toydata$seu
pos <- toydata$pos

adjm <- ProFAST::AddAdj(as.matrix(pos), radius.upper = 200)
X <- Seurat::GetAssayData(object = seu, slot = "data", assay = "RNA")
cellembedding <- cellembedding_matrix(
    X = X,
    adjm = adjm
)
print(cellembedding[1:3, 1:3])
```

---

cellembedding_seurat *Perform CAESAR embedding of Cells Using FAST with Spatial Weights*

Description

This function computes cell embedding using the CAESAR framework with FAST for dimensionality reduction and spatial adjacency weights. It integrates variable feature selection and spatial adjacency information to generate low-dimensional representations for cells.

Usage

```
cellembedding_seurat(
  seu,
  adjm,
  assay = NULL,
  slot = "data",
  nfeatures = 2000,
  q = 50,
  reduction.name = "caesar",
  var.features = NULL,
  ...
)
```

Arguments

seu	A Seurat object. The Seurat object should contain gene expression data and be preprocessed with variable features identified.
adjm	A spatial adjacency matrix representing the relationships between cells or spots in spatial transcriptomic data.
assay	A character string specifying which assay to use from the Seurat object. If NULL, the function will use the default assay.
slot	The data slot to use for feature extraction (e.g., "data", "counts"). Default is "data".
nfeatures	The number of features to select for analysis. Default is 2000.
q	An integer specifying the number of dimensions for the reduced embeddings. Default is 50.
reduction.name	A character string specifying the name for the dimensional reduction. Default is "caesar".
var.features	A vector of variable features (genes) to use for the embedding. If NULL, the function will use variable features stored in the Seurat object.
...	Additional arguments passed to 'FAST_run'.

Value

The modified Seurat object with the co-embedding results (cell and gene embeddings) stored in the specified dimensional reduction slot.

See Also

FAST_run for the main FAST dimensionality reduction algorithm.

Examples

```
data(toydata)

seu <- toydata$seu
pos <- toydata$pos

adjm <- ProFAST::AddAdj(as.matrix(pos), radius.upper = 200)
seu <- cellembedding_seurat(
    seu = seu,
    adjm = adjm
)
print(seu)
```

CoUMAP *Co-embedding UMAP for Genes and Cells in a Seurat Object*

Description

This function performs a co-embedding UMAP of both gene and cell embeddings from a Seurat object. It integrates the dimensionality reduction results for genes and cells into a shared UMAP space.

Usage

```
CoUMAP(
  seu,
  reduction = "caesar",
  reduction.name = "caesarUMAP",
  gene.set = NULL,
  slot = "data",
  assay = "RNA",
  seed = 1,
  ...
)
```

Arguments

seu	A Seurat object containing the single-cell RNA-seq data.
reduction	A character string specifying the name of the dimensional reduction to use (e.g., "caesar"). Default is "caesar".
reduction.name	A character string specifying the name of the new dimensional reduction slot in the Seurat object where the co-embedding UMAP will be stored. Default is "caesarUMAP".
gene.set	A character vector specifying the set of genes to include in the co-embedding. If NULL, all genes are used. Default is NULL.
slot	A character string specifying the slot in the Seurat object to use for gene expression data. Default is "data".
assay	A character string specifying the assay to use. Default is "RNA".
seed	An integer specifying the random seed for reproducibility. Default is 1.
...	Additional arguments passed to scater::calculateUMAP.

Details

The function extracts the embeddings for both genes and cells from the specified dimensional reduction, combines them, and computes a UMAP embedding. The resulting co-embedding UMAP is stored in a new dimensional reduction slot in the Seurat object.

Value

A modified Seurat object with the co-embedding UMAP stored in the specified `reduction.name` slot.

See Also

`calculateUMAP` for UMAP calculation.

Examples

```
data(toydata)

seu <- toydata$seu

seu <- CoUMAP(seu, gene.set = rownames(seu))
print(seu)
```

CoUMAP.plot *Plot Co-embedding UMAP for Genes and Cells*

Description

This function generates a UMAP plot for co-embedding genes and cells in a Seurat object. It allows for customization of point colors, shapes, and text labels, and can display both gene and cell embeddings in the same plot.

Usage

```
CoUMAP.plot(
  seu,
  reduction = "caesarUMAP",
  gene_txtdata = NULL,
  ident = NULL,
  xy_name = reduction,
  dims = c(1, 2),
  cols = NULL,
  shape_cg = c(1, 5),
  pt_size = 1,
  pt_text_size = 5,
  base_size = 16,
  base_family = "serif",
  legend.point.size = 5,
  legend.key.size = 1.5,
  alpha = 0.8
)
```

Arguments

seu	A Seurat object containing the co-embedding data.
reduction	A character string specifying the name of the dimensional reduction to use for the UMAP plot. Default is "caesarUMAP".
gene_txtdata	A data frame containing gene names and labels to display as text on the plot. If NULL, no text labels are shown. Default is NULL.
ident	A character string specifying the column name in the Seurat object's meta-data that contains cell type or cluster labels. If NULL, the default identities (Idents(seu)) are used. Default is NULL.
xy_name	A character string specifying the prefix for the UMAP axes. Default is the value of reduction.
dims	A numeric vector of length 2 specifying which dimensions to plot. Default is c(1, 2).
cols	A named vector of colors for clusters. If NULL, default colors are generated. Default is NULL.
shape_cg	A numeric vector of length 2 specifying the shapes for cells and genes. Default is c(1, 5).
pt_size	Numeric, specifying the size of the points for cells and genes. Default is 1.
pt_text_size	Numeric, specifying the size of the text labels for genes. Default is 5.
base_size	Numeric, specifying the base font size for the plot. Default is 16.
base_family	Character string specifying the font family for the plot. Default is "serif".
legend.point.size	Numeric, specifying the size of the points in the legend. Default is 5.
legend.key.size	Numeric, specifying the size of the legend keys. Default is 1.5.
alpha	Numeric, specifying the transparency level for cell points. Default is 0.8.

Details

The function creates a UMAP plot that shows both gene and cell embeddings. Gene embeddings can be optionally labeled with text, and the plot can be customized with different colors, shapes, and sizes.

Value

A ggplot object representing the UMAP plot.

See Also

CoUMAP for obtain co-embedding UMAP.

Examples

```
data(toydata)

seu <- toydata$seu

seu <- CoUMAP(seu, gene.set = rownames(seu))
CoUMAP.plot(seu)
```

`find.sig.genes` *Identify Signature Genes for Each Cell Type*

Description

This function identifies signature genes for each cell type or cell group in a Seurat object using a co-embedding distance-based approach. It computes the average expression and distance metrics for each gene across different groups, while also considering expression proportions.

Usage

```
find.sig.genes(
  seu,
  distce.assay = "distce",
  ident = NULL,
  expr.prop.cutoff = 0.1,
  assay = NULL,
  genes.use = NULL
)
```

Arguments

`seu`	A Seurat object containing gene expression data.
`distce.assay`	A character string specifying the assay that contains the distance matrix or distance-related data. Default is "distce".
`ident`	A character string specifying the column name in the ‘meta.data‘ slot of the Seurat object used to define the identities (clusters or cell groups). If ‘NULL‘, the default identities (‘Idents(seu)‘) will be used. Default is ‘NULL‘.
`expr.prop.cutoff`	A numeric value specifying the minimum proportion of cells that must express a gene for it to be considered. Default is 0.1.
`assay`	A character string specifying the assay to use for expression data. If ‘NULL‘, the default assay of the Seurat object will be used. Default is ‘NULL‘.
`genes.use`	A character vector specifying the genes to use for the analysis. If ‘NULL‘, all genes in the ‘distce.assay‘ assay will be used. Default is ‘NULL‘.

Value

A list where each element corresponds to a cell group and contains a data frame with the following columns:

`distance`	The mean distance of the gene across the cells in the group.
`expr.prop`	The proportion of cells in the group expressing the gene.
`expr.prop.others`	The proportion of cells in other groups expressing the gene.
`label`	The identity label of the cell group.
`gene`	The gene name.

See Also

None

Examples

```
data(toydata)

seu <- toydata$seu

seu <- ProFAST::pdistance(seu, reduction = "caesar")
sglist <- find.sig.genes(
    seu = seu
)
str(sglist)
```

---

getneighborhood_fastcpp

getneighborhood_fast

Description

an efficient function to find the neighborhood based on the matrix of position and a pre-defined cutoff

Usage

```
getneighborhood_fastcpp(x, radius)
```

Arguments

`x`	is a n-by-2 matrix of position.
`radius`	is a threashold of Euclidean distance to decide whether a spot is an neighborhood of another spot. For example, if the Euclidean distance between spot A and B is less than cutoff, then A is taken as the neighbourhood of B.

Value

A sparse matrix containing the neighbourhood

Human_HK_genes *Human housekeeping genes database*

Description

Human housekeeping genes database.

Details

This data is a data.frame and include the Human housekeeping genes information in the columns named "Gene" and "Ensembl".

Intsg *Integrate Signature Genes Across Datasets*

Description

This function integrates signature genes across different datasets by identifying common genes that meet specific criteria. It filters out mitochondrial and ribosomal genes, allows for the exclusion of genes based on expression proportion, and supports weighting gene selection by cell type ratios.

Usage

```
Intsg(
  sg_List,
  ntop,
  ct_ratio = NULL,
  expr.prop.cutoff = 0.1,
  species = "hm",
  rm_mito_ribo = TRUE,
  ratio_lower_bound = 0
)
```

Arguments

sg_List	A list of signature gene lists for different datasets. Each element in the list should be a named list where the names correspond to cell types, and each cell type contains a data frame with gene information.
ntop	An integer specifying the maximum number of top genes to retain for each cell type.
ct_ratio	A list of numeric vectors specifying the ratio of cells for each cell type in the datasets. If NULL, no weighting is applied. Default is NULL.

`expr.prop.cutoff`
: A numeric value specifying the minimum expression proportion required for a gene to be considered. Default is 0.1.

`species`
: A character string specifying the species, either "hm" (human) or "ms" (mouse). Default is "hm".

`rm_mito_ribo`
: Logical, indicating whether to remove mitochondrial and ribosomal genes from the signature gene list. Default is `TRUE`.

`ratio_lower_bound`
: A numeric value specifying the lower bound for the cell type ratio. Only cell types with a ratio above this bound are considered. Default is 0.0.

Value

A named list where each element corresponds to a cell type and contains the integrated list of top signature genes.

Examples

```
data(toydata)

seu <- toydata$seu

seu <- ProFAST::pdistance(seu, reduction = "caesar")
sglist <- find.sig.genes(seu = seu)
top2sgs <- Intsg(list(sglist), ntop = 2)
print(top2sgs)

top2intsgs <- Intsg(list(sglist, sglist), ntop = 2)
```

`marker.select` *Select Marker Genes from a signature gene list Based on Expression Proportion and Overlap Criteria*

Description

This function selects marker genes for each cluster or cell type based on expression proportion, with options to remove mitochondrial and ribosomal genes, limit the maximum number of top marker genes, and control the overlap between markers across clusters.

Usage

```
marker.select(
  ref_sig_list,
  expr.prop.cutoff = 0.1,
  ntop.max = 200,
  overlap.max = 1,
  rm_mito_ribo = FALSE,
```

```
  species = "ms"
)
```

Arguments

ref_sig_list	A list where each element corresponds to a cluster or cell type. Each element should be a data frame containing at least two columns: gene (the gene names) and expr.prop (the proportion of cells expressing each gene). Generally, it is the output of function find.sig.genes.
expr.prop.cutoff	A numeric value specifying the minimum proportion of cells that must express a gene for it to be considered. Default is 0.1.
ntop.max	An integer specifying the maximum number of top marker genes to be selected. Default is 200.
overlap.max	An integer specifying the maximum allowable overlap of marker genes across clusters. If a gene appears in more than overlap.max clusters, it will be excluded. Default is 1.
rm_mito_ribo	Logical, indicating whether to remove mitochondrial and ribosomal genes from the marker gene list. Default is FALSE.
species	A character string specifying the species for mitochondrial and ribosomal gene detection. Options are "ms" for mouse or "hs" for human. Default is "ms".

Value

A list where each element corresponds to a cluster and contains the selected marker genes. If no markers are found, a message is printed and NULL is returned.

See Also

find.sig.genes for signature gene list.

Examples

```
data(toydata)

seu <- toydata$seu

seu <- ProFAST::pdistance(seu, reduction = "caesar")
sglist <- find.sig.genes(seu = seu)

markers <- marker.select(sglist, expr.prop.cutoff = 0.1, overlap.max = 1)
print(markers)
```

markerList2mat *Convert Marker List to a Weighted Matrix*

Description

This function converts a list of marker genes for different cell types or clusters into a matrix, where rows represent cell types and columns represent marker genes. The matrix contains the frequency of each marker gene across the different cell types.

Usage

```
markerList2mat(markerList)
```

Arguments

markerList	A list where each element contains marker genes for different cell types or clusters. Each element of the list is a named list, where the names are the cell types and the values are the marker genes for that cell type. Generally, it is a list of the output of function `marker.select`.

Value

A matrix with rows representing cell types and columns representing unique marker genes. The values in the matrix represent the frequency of each gene as a marker in the corresponding cell type.

See Also

`marker.select` for select markers. `find.sig.genes` for signature gene list.

Examples

```
data(toydata)

markers <- toydata$markers

marker.freq <- markerList2mat(list(markers))
print(marker.freq)
```

Mouse_HK_genes *Mouse housekeeping genes database*

Description

Mouse housekeeping genes database.

Details

This data is a data.frame and include the mouse housekeeping genes information in the columns named "Gene" and "Ensembl".

SigScore *Calculate Signature Score for Cell Clusters*

Description

This function calculates a signature score for cell clusters based on the embeddings of cells and genes in a Seurat object. The score measures how well the genes linked to specific clusters are separated in the embedding space. The function also supports returning a weighted mean score across all clusters.

Usage

```
SigScore(seu, reduction, label, gclink, return.mean = TRUE)
```

Arguments

seu	A Seurat object containing the single-cell RNA-seq data.
reduction	A character string specifying the name of the dimensional reduction to use (e.g., "caesar").
label	A character string specifying the column name in the Seurat object's metadata that contains the cell type or cluster labels.
gclink	A data frame with two columns: 'gene' and 'cluster', where 'gene' corresponds to gene names and 'cluster' corresponds to the associated cell cluster.
return.mean	Logical, indicating whether to return the weighted mean signature score across all clusters. If FALSE, returns the score for each gene-cluster pair. Default is TRUE.

Details

The function computes the distance between gene embeddings and cell embeddings using a custom distance metric. It then calculates a score for each gene-cluster pair by evaluating how well the genes associated with a specific cluster are ordered in proximity to the cells of that cluster. The score is normalized between 0 and 1, where 1 indicates perfect separation.

Value

If `return.mean = TRUE`, a numeric value representing the weighted mean signature score across all clusters. If `return.mean = FALSE`, a numeric vector containing the signature score for each gene-cluster pair.

Examples

```
library(Seurat)
data(toydata)

seu <- toydata$seu

seu$cluster <- Idents(seu)

gclink <- data.frame(
    gene = c(
        "FBLN1", "CCDC80", "LYPD3", "MLPH", "HOXD9", "EGFL7",
        "HAVCR2", "IGSF6", "KRT5", "KRT6B", "CD79A", "DERL3",
        "CAV1", "AVPR1A", "CD3G", "CD3D"
    ),
    cluster = c(
        "CAFs", "CAFs", "Cancer Epithelial", "Cancer Epithelial",
        "Endothelial", "Endothelial", "Myeloid", "Myeloid",
        "Normal Epithelial", "Normal Epithelial", "Plasmablasts",
        "Plasmablasts", "PVL", "PVL", "T-cells", "T-cells"
    )
)

score <- SigScore(
    seu, reduction = "caesar", label = "cluster", gclink = gclink,
    return.mean = TRUE
)
print(score)

score <- SigScore(
    seu, reduction = "caesar", label = "cluster",
    gclink = gclink, return.mean = FALSE
)
print(score)
```

---

toydata *A toy dataset to run examples*

Description

A list containing the following components:

Usage

```
data(toydata)
```

Format

A list with five components.

- seu: A Seurat object, which is a subset of Xenium breast cancer section.
- pos: A data frame containing the location information of spots. Each row corresponds to a spatial spot, with columns for x and y coordinates.
- pathway_list: A list containing two pathways of intersecting. This list provides information about gene sets or pathways that are relevant to the dataset. Each element in the list is a character vector of gene names.
- markers: A list of marker genes for each cell type. This list maps cell types to their respective marker genes, which can be used for cell type annotation.
- imgf: A matrix containing histology image features. Each row corresponds to a spatial spot, and each column represents a different feature extracted from the histology image.

Index