Package: cellchatr (via r-universe)

November 14, 2026

Title Fast Ligand-Receptor Interaction Screening for Single-Cell Data Using Rust

Version 0.99.1

Description Fast identification of over-expressed genes and ligand-receptor interactions in single-cell RNA-seq data, the pre-processing steps of cell-cell communication analysis tools such as CellChat. Works directly on SummarizedExperiment and SingleCellExperiment objects and accepts any ligand-receptor table (for example the CellChatDB interaction table). The per-gene, per-cell-group Wilcoxon rank-sum tests are implemented in Rust via extendr and run in parallel with Rayon.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/rextendr/version 0.4.2

SystemRequirements Cargo (Rust's package manager), rustc >= 1.65.0, xz

URL https://github.com/mayankgandhi13/cellchatr

BugReports https://github.com/mayankgandhi13/cellchatr/issues

biocViews SingleCell, GeneExpression, Transcriptomics, DifferentialExpression, Network, Software

Depends R (>= 4.4.0)

Imports methods, S4Vectors, SummarizedExperiment

Suggests SingleCellExperiment, Matrix, BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

Config/pak/sysreqs xz-utils zlib1g-dev libclang-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-10-01 01:52:36 UTC

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

RemoteRef HEAD

RemoteSha e315f0b33c2c95e25ae60f30a82488972986578c

Contents

findOverexpressedGenes 2
identifyInteractions 3
matchLRPairs 4
rust_match_lr_pairs 5
rust_subset_genes 6
rust_wilcoxon_filter 6

Index **8**

`findOverexpressedGenes`

Find over-expressed genes per cell group

Description

Runs a one-sided ("greater") Wilcoxon rank-sum test for every gene and every cell group, comparing cells of that group against all other cells. A gene is reported if it is significantly higher in at least one group. Tests are run in parallel in Rust (Rayon), using the normal approximation with tie correction.

Usage

```
findOverexpressedGenes(x, ...)

## S4 method for signature 'ANY'
findOverexpressedGenes(x, labels, lrdb = NULL, pval_threshold = 0.05)

## S4 method for signature 'SummarizedExperiment'
findOverexpressedGenes(x, labels, assay.type = "logcounts", ...)
```

Arguments

`x`	A numeric gene-by-cell matrix (dense `matrix` or any object coercible with `as.matrix()`, such as a sparse `dgCMatrix`) with gene symbols as row names, or a SummarizedExperiment::SummarizedExperiment (including `SingleCellExperiment`).
`...`	Further arguments passed to the matrix method.
`labels`	Cell group labels. Either a vector with one entry per cell (column of `x`) or, for `SummarizedExperiment` input, the name of a `colData(x)` column holding the labels.

lrdb	Optional ligand-receptor table (see matchLRPairs()). If supplied, only genes appearing as a ligand or receptor are tested, which also keeps memory low when x is sparse.
pval_threshold	P-value cutoff for significance.
assay.type	For SummarizedExperiment input, the assay to use. Normalized expression (e.g. "logcounts") is recommended.

Value

A character vector of over-expressed gene names.

Examples

```
counts <- matrix(c(rep(10, 50), rep(0, 50), rep(1, 100)),
  nrow = 2, byrow = TRUE,
  dimnames = list(c("TGFB1", "GAPDH"), NULL))
labels <- c(rep("A", 50), rep("B", 50))
findOverexpressedGenes(counts, labels)

# SummarizedExperiment / SingleCellExperiment input
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(logcounts = counts),
  colData = S4Vectors::DataFrame(cell_type = labels))
findOverexpressedGenes(se, labels = "cell_type")
```

---

identifyInteractions *Identify over-expressed ligand-receptor interactions*

Description

Convenience wrapper running the full workflow: restrict x to genes in the ligand-receptor table, find over-expressed genes per cell group with findOverexpressedGenes(), and return the interactions they take part in with matchLRPairs().

Usage

```
identifyInteractions(x, lrdb, ...)
```

Arguments

x	A numeric gene-by-cell matrix (dense matrix or any object coercible with as.matrix(), such as a sparse dgCMatrix) with gene symbols as row names, or a SummarizedExperiment::SummarizedExperiment (including SingleCellExperiment).
lrdb	Ligand-receptor table, see matchLRPairs().
...	Further arguments passed to findOverexpressedGenes(), such as labels, assay.type and pval_threshold.

Value

A S4Vectors::DataFrame of matching interactions, with the over-expressed genes stored in `metadata(result)$overexpress`

Examples

```
library(SingleCellExperiment)
set.seed(1)
genes <- c("TGFB1", "TGFBR1", "VEGFA", "FLT1", paste0("G", 1:20))
mat <- matrix(rnorm(length(genes) * 60, mean = 1), nrow = length(genes),
  dimnames = list(genes, NULL))
cell_type <- rep(c("Fibroblast", "Endothelial"), each = 30)
mat["TGFB1", cell_type == "Fibroblast"] <- 6
sce <- SingleCellExperiment(assays = list(logcounts = mat),
  colData = DataFrame(cell_type = cell_type))

lrdb <- data.frame(
  interaction_name = c("TGFB1_TGFBR1", "VEGFA_FLT1"),
  ligand = c("TGFB1", "VEGFA"),
  receptor = c("TGFBR1", "FLT1"))

identifyInteractions(sce, lrdb, labels = "cell_type")
```

---

matchLRPairs *Match genes to ligand-receptor interactions*

Description

Keeps the interactions in a ligand-receptor table whose ligand OR receptor is among the supplied genes. The lookup is done with a Rust hash set.

Usage

```
matchLRPairs(genes, lrdb)
```

Arguments

`genes`	Character vector of gene names (typically the output of `findOverexpressedGenes()`).
`lrdb`	A `data.frame` or S4Vectors::DataFrame with one row per interaction and columns `ligand`, `receptor` and `interaction_name`. The `interaction` table of CellChat's `CellChatDB.human` / `CellChatDB.mouse` has this layout, but any ligand-receptor resource can be used. Ligands and receptors are matched by exact name, so multi-subunit complexes are only matched if the complex name itself is in `genes`.

Value

A S4Vectors::DataFrame holding the matching rows of `lrdb`, with all of its columns.

Examples

```
lrdb <- data.frame(
  interaction_name = c("TGFB1_TGFBR1", "MYC_MYCBP", "VEGFA_FLT1"),
  ligand = c("TGFB1", "MYC", "VEGFA"),
  receptor = c("TGFBR1", "MYCBP", "FLT1"))
matchLRPairs(c("TGFB1", "FLT1"), lrdb)
```

---

rust_match_lr_pairs *Match overexpressed genes against CellChatDB ligand-receptor pairs.*

Description

Match overexpressed genes against CellChatDB ligand-receptor pairs.

Usage

```
rust_match_lr_pairs(overexpressed, lr_ligands, lr_receptors, lr_names)
```

Arguments

`overexpressed`	Character vector of overexpressed gene names.
`lr_ligands`	Character vector of ligands from CellChatDB.
`lr_receptors`	Character vector of receptors from CellChatDB.
`lr_names`	Character vector of interaction names from CellChatDB.

Value

Character vector of matched interaction names.

Complexity

- Time: O(m + r) where m = overexpressed genes, r = LR pairs
- Space: O(m) for HashSet
- R equivalent (%%in%% twice): O(m x r)

Examples

```
rust_match_lr_pairs(
  c("TGFB1", "VEGFA"),
  c("TGFB1", "MYC", "VEGFA"),
  c("TGFBR1", "MYCBP", "FLT1"),
  c("TGFB1_TGFBR1", "MYC_MYCBP", "VEGFA_FLT1")
)
```

rust_subset_genes *Filter expression matrix genes to only those present in CellChatDB.*

Description

Filter expression matrix genes to only those present in CellChatDB.

Usage

```
rust_subset_genes(expr_genes, db_genes)
```

Arguments

expr_genes	Character vector of gene names from expression matrix.
db_genes	Character vector of gene names from CellChatDB.

Value

Character vector of genes present in both inputs.

Complexity

- Time: O(n + m) where n = db_genes, m = expr_genes
- Space: O(n) for HashSet
- R equivalent (%%in%%): O(n x m)

Examples

```
rust_subset_genes(
  c("TGFB1", "VEGFA", "GAPDH"),
  c("TGFB1", "VEGFA", "MYC")
)
```

rust_wilcoxon_filter *Identify over-expressed genes per cell type using Wilcoxon rank-sum test.*

Description

For each gene and each cell type, a one-sided ("greater") Wilcoxon rank-sum test compares the cells of that type against all other cells, using the normal approximation with tie correction. A gene is returned if it is significantly higher in at least one cell type.

Usage

```
rust_wilcoxon_filter(counts, labels, gene_names, pval_threshold)
```

Arguments

counts	NumericMatrix of gene expression (genes x cells).
labels	Character vector of cell type labels, one per cell.
gene_names	Character vector of gene names.
pval_threshold	P-value cutoff for significance.

Value

Character vector of over-expressed gene names.

Complexity

- Time: O((g/p) x k x c log c) where g = genes, p = CPU cores, k = cell types, c = cells
- Space: O(c) per gene
- R equivalent: O(g x k x c log c) — serial, no parallelism

Examples

```
counts <- matrix(c(rep(10, 50), rep(0, 50), rep(1, 100)),
  nrow = 2, byrow = TRUE)
labels <- c(rep("A", 50), rep("B", 50))
rust_wilcoxon_filter(counts, labels, c("TGFB1", "GAPDH"), 0.05)
```

Index

findOverexpressedGenes, 2
findOverexpressedGenes(), *3*, *4*
findOverexpressedGenes,ANY-method
        *(*findOverexpressedGenes*)*, 2
findOverexpressedGenes,SummarizedExperiment-method
        *(*findOverexpressedGenes*)*, 2

identifyInteractions, 3

matchLRPairs, 4
matchLRPairs(), *3*

rust_match_lr_pairs, 5
rust_subset_genes, 6
rust_wilcoxon_filter, 6

S4Vectors::DataFrame, *4*
SummarizedExperiment::SummarizedExperiment,
        2, *3*