Package 'markerpen'

March 17, 2021

Type Package

Title Marker Gene Detection via Penalized Principal Component Analysis

Version 0.1.1

Date 2021-03-14

Author Yixuan Qiu, Jiebiao Wang, Jing Lei, and Kathryn Roeder

Maintainer Yixuan Qiu <yixuan.qiu@cos.name>

Description Implementation of the 'MarkerPen' algorithm, short for marker gene detection via penalized principal component analysis, described in the paper by Qiu, Wang, Lei, and Roeder (2020, <doi:10.1101/2020.11.07.373043>). 'MarkerPen' is a semi-supervised algorithm for detecting marker genes by combining prior marker information with bulk transcriptome data.

License GPL

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.1), RSpectra, stats

LinkingTo Rcpp, RcppEigen, RSpectra

Suggests knitr, rmarkdown, prettydoc, scales

SystemRequirements C++11

VignetteBuilder knitr, rmarkdown

RoxygenNote 7.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-03-17 00:30:02 UTC

R topics documented:

`gene_mapping` *Mapping gene names to Ensembl IDs*

Description

A data set showing the mapping between gene names and Ensembl gene IDs, derived from the **EnsDb.Hsapiens.v79** Bioconductor package.

Usage

```
gene_mapping
```

Format

A data frame with 59074 rows and 2 variables:

ensembl Ensembl gene IDs

name corresponding gene names

Source

https://bioconductor.org/packages/release/data/annotation/html/EnsDb.Hsapiens.v79.html

---

`pca_pen` *Penalized Principal Component Analysis for Marker Gene Selection*

Description

This function solves the optimization problem

$$\min \quad -\mathrm{tr}(SX) + \lambda p(X),$$

$$s.t. \quad O \preceq X \preceq I, \quad X \geq 0, \quad \text{and} \quad \mathrm{tr}(X) = 1,$$

where $O \preceq X \preceq I$ means all eigenvalues of X are between 0 and 1, $X \geq 0$ means all elements of X are nonnegative, and $p(X)$ is a penalty function defined in the article (see the **References** section).

Usage

```
pca_pen(
  S,
  gr,
  lambda,
  w = 1.5,
  alpha = 0.01,
  maxit = 1000,
  eps = 1e-04,
  verbose = 0
)
```

Arguments

S	The sample correlation matrix of gene expression.
gr	Indices of genes that are treated as markers in the prior information.
lambda	Tuning parameter to control the sparsity of eigenvectors.
w	Tuning parameter to control the weight on prior information. Larger w means genes not in the prior list are less likely to be selected as markers.
alpha	Step size of the optimization algorithm.
maxit	Maximum number of iterations.
eps	Tolerance parameter for convergence.
verbose	Level of verbosity.

Value

A list containing the following components:

projection The estimated projection matrix.

evecs The estimated eigenvectors.

niter Number of iterations used in the optimization process.

err_v The optimization error in each iteration.

References

Qiu, Y., Wang, J., Lei, J., & Roeder, K. (2020). Identification of cell-type-specific marker genes from co-expression patterns in tissue samples.

Examples

```
set.seed(123)
n = 200  # Sample size
p = 500  # Number of genes
s = 50   # Number of true signals

# The first s genes are true markers, and others are noise
Sigma = matrix(0, p, p)
```

```
Sigma[1:s, 1:s] = 0.9
diag(Sigma) = 1

# Simulate data from the covariance matrix
x = matrix(rnorm(n * p), n) %*% chol(Sigma)

# Sample correlation matrix
S = cor(x)

# Indices of prior marker genes
# Note that we have omitted 10 true markers, and included 10 false markers
gr = c(1:(s - 10), (s + 11):(s + 20))

# Run the algorithm
res = pca_pen(S, gr, lambda = 0.1, verbose = 1)

# See if we can recover the true correlation structure
image(res$projection, asp = 1)
```

`refine_markers` *Marker Gene Selection via Penalized Principal Component Analysis*

Description

This function refines a prior marker gene list by combining information from bulk tissue data, based on the penalized principal component analysis. The current implementation computes on one cell type at a time. To get marker genes for multiple cell types, call this function iteratively.

Usage

```
refine_markers(
  mat_exp,
  range,
  markers,
  lambda,
  w = 1.5,
  thresh = 0.001,
  alpha = 0.01,
  maxit = 1000,
  eps = 1e-04,
  verbose = 0
)
```

Arguments

`mat_exp`	The gene expression matrix in the original scale (not logarithm-transformed), with rows standing for observations and columns for genes. The matrix should include gene names as column names.

range	A character vector of gene names, representing the range of genes in which markers are sought.
markers	A character vector of gene names giving the prior marker gene list.
lambda	A tuning parameter to control the number of selected marker genes. A larger value typically means a smaller number of genes.
w	Tuning parameter to control the weight on prior information. Larger w means genes not in the prior list are less likely to be selected as markers.
thresh	Below this threshold small factor loadings are treated as zeros.
alpha	Step size of the optimization algorithm.
maxit	Maximum number of iterations.
eps	Tolerance parameter for convergence.
verbose	Level of verbosity.

Value

A list containing the following components:

spca The sparse PCA result as in `pca_pen()`.

markers A character vector of selected markers genes.

markers_coef The estimated factor loadings for the associated genes.

References

Qiu, Y., Wang, J., Lei, J., & Roeder, K. (2020). Identification of cell-type-specific marker genes from co-expression patterns in tissue samples.

Examples

```
# Data used in the vignette
load(system.file("examples", "gene_expr.RData", package = "markerpen"))
load(system.file("examples", "published_markers.RData", package = "markerpen"))
load(system.file("examples", "markers_range.RData", package = "markerpen"))

# Get expression matrix - rows are observations, columns are genes
ind = match(rownames(dat), markerpen::gene_mapping$name)
ind = na.omit(ind)
ensembl = markerpen::gene_mapping$ensembl[ind]
mat_exp = t(dat[markerpen::gene_mapping$name[ind], ])
colnames(mat_exp) = ensembl

# We compute the marker genes for two cell types with a reduced problem size
# See the vignette for the full example

# Markers for astrocytes
set.seed(123)
search_range = intersect(markers_range$astrocytes, ensembl)
search_range = sample(search_range, 300)
prior_markers = intersect(pub_markers$astrocytes, search_range)
```

```
ast_re = refine_markers(
    mat_exp, search_range, prior_markers,
    lambda = 0.35, w = 1.5, maxit = 500, eps = 1e-3, verbose = 0
)
# Remove selected markers from the expression matrix
mat_rest = mat_exp[, setdiff(colnames(mat_exp), ast_re$markers)]

# Markers for microglia
search_range = intersect(markers_range$microglia, ensembl)
search_range = sample(search_range, 300)
prior_markers = intersect(pub_markers$microglia, search_range)
mic_re = refine_markers(
    mat_exp, search_range, prior_markers,
    lambda = 0.35, w = 1.5, maxit = 500, eps = 1e-3, verbose = 0
)

# Refined markers
markers_re = list(astrocytes = ast_re$markers,
                  microglia  = mic_re$markers)
# Visualize the correlation matrix
cor_markers = cor(mat_exp[, unlist(markers_re)])
image(cor_markers, asp = 1)

# Post-process the selected markers
# Pick the first 20 ordered markers
markers_ord = sort_markers(cor_markers, markers_re)
markers_ord = lapply(markers_ord, head, n = 20)
# Visualize the correlation matrix
image(cor(mat_exp[, unlist(markers_ord)]), asp = 1)
```

`sort_markers` *Post-processing Selected Marker Genes*

Description

This function reorders the selected marker genes using information of the sample correlation matrix.

Usage

```
sort_markers(corr, markers)
```

Arguments

`corr`	The sample correlation matrix, whose row and column names are gene names.
`markers`	A list of marker genes. Each component of the list is a vector of marker gene names corresponding to a cell type. All the gene names in this list must appear in the row/column names of `corr`.

Value

A list that has the same structure as the input `markers` argument, with the elements in each component reordered. See the example in `refine_markers()`.

Index