Package ‘GeneNMF’

February 29, 2024

Type Package

Title Non-Negative Matrix Factorization for Single-Cell Omics

Version 0.4.0

Description A collection of methods to extract gene programs from single-cell gene expression data using non-negative matrix factorization (NMF). ‘GeneNMF’ contains functions to directly interact with the 'Seurat' toolkit and derive interpretable gene program signatures.

biocViews

Depends R (>= 4.3.0)
Imports RcppML, NMF, stats, Seurat (>= 4.3.0), cluster, pheatmap, viridis
Suggests knitr, markdown, fgsea, dplyr, msigdb
VignetteBuilder knitr

URL https://github.com/carmonalab/GeneNMF

BugReports https://github.com/carmonalab/GeneNMF/issues

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

NeedsCompilation no

Author Massimo Andreatta [aut, cre] (<https://orcid.org/0000-0002-8036-2647>), Santiago Carmona [aut] (<https://orcid.org/0000-0002-2495-0671>)

Maintainer Massimo Andreatta <massimo.andreatta@unil.ch>

Repository CRAN

Date/Publication 2024-02-29 12:22:40 UTC
findVariableFeatures_wfilters

**Find variable features**

**Description**

Select highly variable genes (HVG) from an expression matrix. Genes from a blocklist (e.g. cell cycling genes, mitochondrial genes) can be excluded from the list of variable genes, as well as genes with very low or very high average expression.

**Usage**

```r
findVariableFeatures_wfilters(
  obj,  
  nfeatures = 2000,  
  genesBlockList = NULL,  
  min.exp = 0.01,  
  max.exp = 3
)
```

**Arguments**

- `obj`: A Seurat object containing an expression matrix
- `nfeatures`: Number of top HVG to be returned
- `genesBlockList`: Optionally takes a vector or list of vectors of gene names. These genes will be ignored for HVG detection. This is useful to mitigate effect of genes associated with technical artifacts or batch effects (e.g. mitochondrial, heat-shock response). If set to ‘NULL’ no genes will be excluded
- `min.exp`: Minimum average normalized expression for HVG. If lower, the gene will be excluded
- `max.exp`: Maximum average normalized expression for HVG. If higher, the gene will be excluded
**getValue**

Returns the input Seurat object `obj` with the calculated highly variable features accessible through `VariableFeatures(obj)`

**Examples**

```r
data(sampleObj)
sampleObj <- findVariableFeatures_wfilters(sampleObj, nfeatures=100)
```

---

**getDataMatrix**

*Extract data matrix from Seurat object*

**Description**

Get the gene expression matrix from a Seurat object, optionally centered and/or subset on highly variable genes

**Usage**

```r
data(getDataMatrix(
  obj,
  assay = "RNA",
  slot = "data",
  hvg = NULL,
  do_centering = TRUE
))
```

**Arguments**

- `obj` Seurat object
- `assay` Get data matrix from this assay
- `slot` Get data matrix from this slot (=layer)
- `hvg` List of variable genes to subset the matrix. If NULL, uses all genes
- `do_centering` Whether to center the data matrix

**Value**

Returns a sparse data matrix (cells per genes), subset according to the given parameters

**Examples**

```r
data(sampleObj)
matrix <- getDataMatrix(sampleObj)
```
**getMetaPrograms**

Extract consensus gene programs (meta-programs)

**Description**

Run it over a list of NMF models obtained using `multiNMF`; it will determine gene programs that are consistently observed across samples and values of k.

**Usage**

```r
getMetaPrograms(
  nmf.res,
  method = 0.5,
  max.genes = 200,
  hclust.method = "ward.D2",
  nprograms = 10,
  min.confidence = 0.2,
  remove.empty = TRUE
)
```

**Arguments**

- `nmf.res`: A list of NMF models obtained from `multiNMF`
- `method`: Parameter passed to `extractFeatures` to obtain top genes for each program
- `max.genes`: Max number of genes for each programs
- `hclust.method`: Method to build similarity tree between individual programs
- `nprograms`: Total number of meta-programs
- `min.confidence`: Percentage of programs in which a gene is seen (out of programs in the corresponding program tree branch/cluster), to be retained in the consensus meta-programs
- `remove.empty`: Whether to remove meta-programs with no genes above confidence threshold

**Value**

Returns a list with i) 'metaprograms.genes' top genes for each meta-program; ii) 'metaprograms.metrics' dataframe with meta-programs statistics: a) freq. of samples where the MP is present, b) average silhouette width, c) mean Jaccard similarity, d) number of genes in MP, e) number of gene programs in MP; iii) 'programs.jaccard': matrix of Jaccard similarities between meta-programs; iv) 'programs.tree': hierarchical clustering of meta-programs (hclust tree); v) 'programs.clusters': meta-program identity for each program.
getNMFgenes

Examples

library(Seurat)
data(sampleObj)
geneNMF_programs <- multiNMF(list(sampleObj), k=5)
geneNMF_metaprograms <- getMetaPrograms(geneNMF_programs, nprograms=3)

getNMFgenes

Get list of genes for each NMF program

Description

Run it over a list of NMF models obtained using multiNMF()

Usage

getNMFgenes(nmf.res, method = 0.5, max.genes = 200)

Arguments

nmf.res A list of NMF models obtained using multiNMF()
method Parameter passed to extractFeatures to obtain top genes for each program. When 'method' is a number between 0 and 1, it indicates the minimum relative basis contribution above which the feature is selected, i.e. how specific is a gene for a given program.
max.genes Max number of genes for each program

Value

Returns a list of top genes for each gene program found by multiNMF()

Examples

library(Seurat)
data(sampleObj)
geneNMF_programs <- multiNMF(list(sampleObj), k=5)
geneNMF_genes <- getNMFgenes(geneNMF_programs)
multiNMF

Run NMF on a list of Seurat objects

Description

Given a list of Seurat objects, run non-negative matrix factorization on each sample individually, over a range of target NMF components (k).

Usage

```r
multiNMF(
    obj.list,
    assay = "RNA",
    slot = "data",
    k = 5:6,
    hvg = NULL,
    nfeatures = 2000,
    L1 = c(0, 0),
    min.exp = 0.01,
    max.exp = 3,
    do_centering = TRUE,
    min.cells.per.sample = 10,
    hvg.blocklist = NULL,
    seed = 123
)
```

Arguments

- `obj.list`: A list of Seurat objects
- `assay`: Get data matrix from this assay
- `slot`: Get data matrix from this slot (=layer)
- `k`: Number of target components for NMF (can be a vector)
- `hvg`: List of pre-calculated variable genes to subset the matrix. If hvg=NULL it calculates them automatically
- `nfeatures`: Number of HVG, if calculate_hvg=TRUE
- `L1`: L1 regularization term for NMF
- `min.exp`: Minimum average log-expression value for retaining genes
- `max.exp`: Maximum average log-expression value for retaining genes
- `do_centering`: Whether to center the data matrix
- `min.cells.per.sample`: Minimum number of cells per sample (smaller samples will be ignored)
plotMetaPrograms

hvg.blocklist  Optionally takes a vector or list of vectors of gene names. These genes will be ignored for HVG detection. This is useful to mitigate effect of genes associated with technical artifacts and batch effects (e.g. mitochondrial), and to exclude TCR and BCR adaptive immune (clone-specific) receptors. If set to ‘NULL’ no genes will be excluded

seed  Random seed

Value

Returns a list of NMF programs, one for each sample and for each value of ‘k’. The format of each program in the list follows the structure of nmf factorization models.

Examples

library(Seurat)
data(sampleObj)
geneNMF_programs <- multiNMF(list(sampleObj), k=5)

---

plotMetaPrograms  Visualizations for meta-programs

Description

Generates a clustered heatmap for meta-program similarities (by Jaccard index). This function is intended to be run on the object generated by getMetaPrograms, which contains a pre-calculated tree of pairwise similarities between clusters (as a 'hclust' object).

Usage

plotMetaPrograms(
  mp.res,
  jaccard.cutoff = c(0, 0.8),
  scale = "none",
  palette = viridis(100, option = "A", direction = -1),
  annotation_colors = NULL,
  main = "Clustered Heatmap",
  show_rownames = FALSE,
  show_colnames = FALSE,
  ...
)

Arguments

mp.res  The meta-programs object generated by getMetaPrograms
jaccard.cutoff  Min and max values for plotting the Jaccard index
scale  Heatmap rescaling (passed to pheatmap as ‘scale’)

runGSEA

palette

Heatmap color palette (passed to pheatmap as 'color')

annotation_colors

Color palette for MP annotations

main

Heatmap title

show_rownames

Whether to display individual program names as rows

show_colnames

Whether to display individual program names as cols

...

Additional parameters for pheatmap

Value

Returns a clustered heatmap of MP similarities, in ggplot2 format

Examples

library(Seurat)
data(sampleObj)
geneNMF_programs <- multiNMF(list(sampleObj), k=5)
geneNMF_metaprograms <- getMetaPrograms(geneNMF_programs, nprograms=3)
plotMetaPrograms(geneNMF_metaprograms)

runGSEA

Run Gene set enrichment analysis

Description

Utility function to run Gene set enrichment analysis (GSEA) against gene sets from MSigDB.

Usage

runGSEA(
    genes,
    universe = NULL,
    category = "H",
    subcategory = NULL,
    species = "Homo sapiens",
    pval.thr = 0.05
)

Arguments

genes

A vector of genes

universe

Background universe of gene symbols (passed on to fgsea::fora)

category

GSEA main category (e.g. "H" or "C5")

subcategory

GSEA subcategory

species

Species for GSEA analysis. For a list of the available species, type msigdb::msigdb_species()

pval.thr

Min p-value to include results
runNMF

Value

Returns a table of enriched gene programs from GSEA

Examples

data(sampleObj)
genesiset <- c("BANK1", "CD22", "CD79A", "CD19", "IGHD", "IGHG3", "IGHM")
gsea_res <- runGSEA(geneset, universe=rownames(sampleObj), category = "C8")

runNMF

Compute NMF as a low-dim embedding for Seurat

Description

Compute NMF embeddings for single-cell dataset, and store them in the Seurat data structure. They can be used as an alternative to PCA for downstream analyses.

Usage

runNMF(
  obj,
  assay = "RNA",
  slot = "data",
  k = 10,
  new.reduction = "NMF",
  seed = 123,
  L1 = c(0, 0),
  hvg = NULL,
  do_centering = TRUE
)

Arguments

obj A seurat object
assay Get data matrix from this assay
slot Get data matrix from this slot (=layer)
k Number of components for low-dim representation
new.reduction Name of new dimensionality reduction
seed Random seed
L1 L1 regularization term for NMF
hvg Which genes to use for the reduction
do_centering Whether to center the data matrix
sampleObj

Value

Returns a Seurat object with a new dimensionality reduction (NMF)

Examples

data(sampleObj)
sampleObj <- runNMF(sampleObj, k=8)

sampleObj

Description

A Seurat object containing single-cell transcriptomes (scRNA-seq) for 50 cells and 20729 genes. Single-cell UMI counts were normalized using a standard log-normalization: counts for each cell were divided by the total counts for that cell and multiplied by 10,000, then natural-log transformed using ‘log1p’.

This a subsample of 25 predicted B cells and 25 predicted NK cells from the large scRNA-seq PBMC dataset published by Hao et al. (doi:10.1016/j.cell.2021.04.048) and available as UMI counts at https://atlas.fredhutch.org/data/nygc/multimodal/pbmc_multimodal.h5seurat

Usage

sampleObj

Format

A sparse matrix of 50 cells and 20729 genes.

Source

Index

* datasets
  sampleObj, 10

extractFeatures, 4, 5

findVariableFeatures_wfilters, 2

dataMatrix, 3

getMetaPrograms, 4, 7

getNMFgenes, 5

multiNMF, 4, 6

nMf, 7

plotMetaPrograms, 7

runGSEA, 8

runNMF, 9

sampleObj, 10