Example_scTypeGSEA

Description

This package is designed to assign cell type labels for each identified cluster in single-cell data. This package uses the "Seurat" R package to do data pre-processing and cell clustering. After clustering the cells, we use differential gene expression analysis to compute the fold-change in gene expression by comparing the cluster profile against all the other identified clusters together. Then we use the Gene Set Enrichment Analysis (GSEA) technique to compute the enrichment (NES and their associated P-value) of marker genes defined for a set of cell types. GSEA analysis provides us with the most statistically relevant cell type for each cluster that is finally assigned to the group.

Installation

You can use following codes to install the package. To use some functions in "fgsea", your R version must be >= 3.6.0.

```
# You may need following codes to install dependent packages.
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("fgsea")
BiocManager::install("MAST")
install.packages("Seurat")

library(devtools)
install_github("cailab-tamu/scTypeGSEA")
```

And then you can use following code to load the package.

```
library(scTypeGSEA)
```

Quick example

Here we use a toy data set "small_RNA" to show our main function "assignCellType" for single cell RNA sequence data. This function can achieve quality control, data pre-process, cluster, get fold changes, do GSEA and label the cell in one step.

```
pbmc_example_res <- assignCellType(small_RNA, min.cells = 1, min.features = 10,
                                   nfeatures = 100, npcs = 10,
                                   dims = 1:10, k.param = 5, resolution = 0.75,
                                   min.pct = 0.25, test.use = "MAST", minSize = 5)
```

```
## The input is not a Seurat object, next to transform gene express count matrix to Seurat object.

## Centering and scaling data matrix

## Computing nearest neighbor graph

## Computing SNN

## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 80
## Number of edges: 2873
##
## Running Louvain algorithm...
```

```
## Maximum modularity in 10 random starts: 0.3564
## Number of communities: 2
## Elapsed time: 0 seconds
## Find marker genes for cluster 0
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
##
## Done!
## Find marker genes for cluster 1
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
##
## Done!
## Do GSEA for cluster 0
## Do GSEA for cluster 1
```

It will return a list with 4 slots. The first slot is a Seurat object.

```
pbmc_example_res$Seurat_obj
```

```
## An object of class Seurat
## 230 features across 80 samples within 1 assay
## Active assay: RNA (230 features)
##  1 dimensional reduction calculated: pca
```

The second slot is a data frame including "cluster-ID", "cell type" and "p-value" for each cell.

```
head(pbmc_example_res$cell_mat)
```

```
##                ClusterID Cell Type  padj
## ATGCCAGAACGACT "1"       "NK_cells" "7.18829936801286e-10"
## CATGGCCTGTGCAT "1"       "NK_cells" "7.18829936801286e-10"
## GAACCTGATGAACC "1"       "NK_cells" "7.18829936801286e-10"
## TGACTGGATTCTCA "1"       "NK_cells" "7.18829936801286e-10"
## AGTCAGACTGCACA "1"       "NK_cells" "7.18829936801286e-10"
## TCTGATACACGTGT "1"       "NK_cells" "7.18829936801286e-10"
```

The third slot is "cluster list". For each cluster, there will be a ranked gene/feature list, which is used to do DSEA. The fourth slot is data frame including "cell type", "NES" and "padj" for each cluster.

```
head(pbmc_example_res$cluster_celltype)
```

```
##          cell type         NES                padj
## Cluster0 "NK_cells"        "2.86436828618051" "7.18829936801286e-10"
## Cluster1 "Dendritic_cells" "2.54600207137584" "6.306799398822e-06"
```

Step by step example

Next is to illustrate this pipeline step by step.

Load data and quality control

Here we will use the data set "pbmc_raw", which has 32738 genes and 2700 cells, to go through this pipeline. The first step is to create Seurat Object, do quality control and then do basic data pre-process, which includes "Normalization", "Scale data", "Find HVG" and "PCA". Here we do "LogNormalize" normalization, "vst" feature selections to select 2000 features and do PCA to get the first 50 PCs. The input can be a Seurat object or a count gene expression matrix.

```
library(scTypeGSEA)
pbmc <- scqc(pbmc_raw, min.cells = 3, min.features = 200, percent.mt = 5,
             normalization.method = "LogNormalize", scale.factor = 10000,
            selection.method = "vst", nfeatures = 2000, npcs = 50)
```

```
## The input is not a Seurat object, next to transform gene express count matrix to Seurat object.
```

```
## Centering and scaling data matrix
```

```
pbmc
```

```
## An object of class Seurat
## 13714 features across 2643 samples within 1 assay
## Active assay: RNA (13714 features)
##  1 dimensional reduction calculated: pca
```

Data pre-process and do cluster

The second step is to cluster data. For single cell RNA sequence data, we will base on "FindClusters" function in the Seurat package.

```
set.seed(47)
pbmc <- doClustering(pbmc, dims = 1:50, k.param = 20, resolution = 0.5)
```

```
## Computing nearest neighbor graph
```

```
## Computing SNN
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 2643
## Number of edges: 201855
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8308
## Number of communities: 7
## Elapsed time: 0 seconds
```

```
head(pbmc@meta.data$seurat_clusters)
```

```
## [1] 0 3 0 4 2 0
## Levels: 0 1 2 3 4 5 6
```

If there are clusters for the cells, you can set "cluster_cell" to add cluster to cells.

```
## don't run
pbmc <- doClustering(pbmc, cluster_cell = Your_cluster_list_for_cell)
```

Get fold changes

Before Gene set enrichment analysis (GSEA), for each cluster, we need a gene rank list. Here we do gene differential expression analysis for each cluster regarding all others to achieve this. The default is "MAST" method. For each cluster, it will return a ranked gene list.

```
cluster_list <- getFC(pbmc, min.pct = 0.25, test.use = "MAST")
```

```
## Find marker genes for cluster 0
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
##
## Done!
## Find marker genes for cluster 1
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
##
## Done!
## Find marker genes for cluster 2
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
```

```
##
## Done!
## Find marker genes for cluster 3
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
##
## Done!
## Find marker genes for cluster 4
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
##
## Done!
## Find marker genes for cluster 5
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
##
## Done!
## Find marker genes for cluster 6
## Assuming data assay in position 1, with name et is log-transformed.
##
## Done!
## Combining coefficients and standard errors
## Calculating log-fold changes
## Calculating likelihood ratio tests
## Refitting on reduced model...
```

```
##
## Done!
```

```
head(cluster_list[[1]])
```

```
##      IL7R      LDHB      CD3D      CD3E     NOSIP       LTB
## 1.2538187 1.1914868 1.1843738 1.0375362 0.9759263 0.9529190
```

Gene set enrichment analysis (GSEA)

After getting rank of gene list for each cluster, compared it with cell type markers database, GSEA method will give us NSF and p-value. By NSF, we can decide the cell type for each cluster. The default database is "PanglaoDB" database. This function will give us cell type for each cluster, with its NSF and padj.

```
cluster_celltype <- doGSEA(cluster_list = cluster_list, db = "PanglaoDB_list",
                           minSize = 15, maxSize = 500)
```

```
## Do GSEA for cluster 0
## Do GSEA for cluster 1
## Do GSEA for cluster 2
## Do GSEA for cluster 3
## Do GSEA for cluster 4
## Do GSEA for cluster 5
## Do GSEA for cluster 6
```

```
head(cluster_celltype)
```

```
##          cell type         NES                padj
## Cluster0 "T_cells"         "2.56054175593529" "5.18028174072102e-06"
## Cluster1 "Neutrophils"     "2.29956534856229" "2.10866623817437e-06"
## Cluster2 "NK_cells"        "2.81922543534295" "9.96125097848532e-18"
## Cluster3 "B_cells_naive"   "2.87624750621789" "1.61417896374571e-13"
## Cluster4 "Dendritic_cells" "2.09699440161317" "5.56854459861125e-05"
## Cluster5 "Dendritic_cells" "2.58621936312799" "7.20051157046717e-10"
```

You can also use your own cell type markers database. Setting "db" to be a path to the new database that to be used, which is a list of cell types with their marker genes. The file must be in 'rds' format.

```
## Don't run
cluster_celltype <- doGSEA(cluster_list = cluster_list, db = "path/to/your/rdsfile")
```

Label the cell type

The last step is to add label cell type to our Seurat object.

```
pbmc_res <- labelCelltype(pbmc, cluster_celltype)
```

It will return a list with 2 slots. The first slot is a Seurat object.

```
pbmc_res$obj
```

```
## An object of class Seurat
## 13714 features across 2643 samples within 1 assay
## Active assay: RNA (13714 features)
##  1 dimensional reduction calculated: pca
```

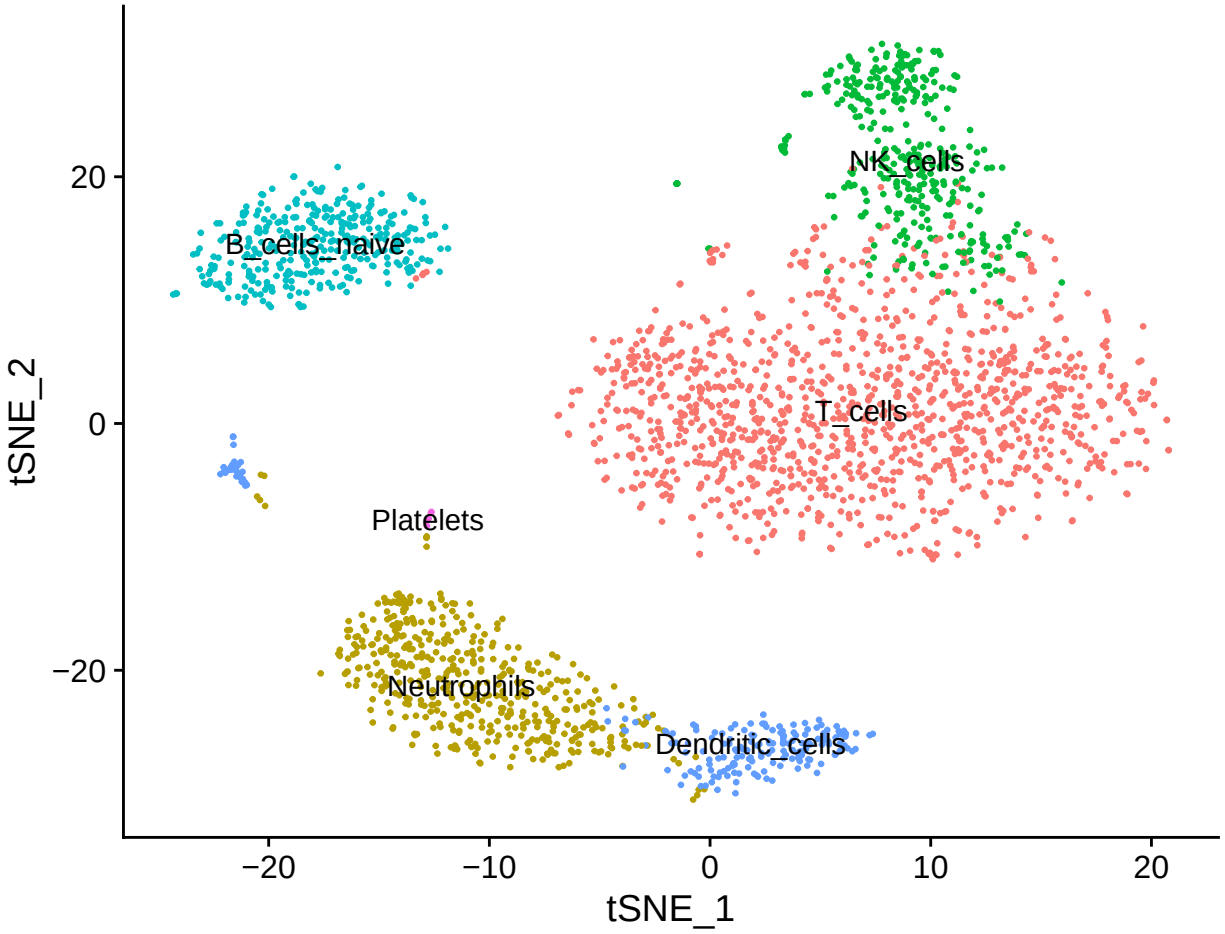
The second slot is a data frame including "cluster-ID", "cell type" and "p-value" for each cell.

```
head(pbmc_res$cell_mat)
```

```
##                ClusterID Cell Type         padj
## AAACATACAACCAC "1"       "T_cells"         "5.18028174072102e-06"
## AAACATTGAGCTAC "4"       "B_cells_naive"   "1.61417896374571e-13"
## AAACATTGATCAGC "1"       "T_cells"         "5.18028174072102e-06"
## AAACCGTGCTTCCG "5"       "Dendritic_cells" "5.56854459861125e-05"
## AAACCGTGTATGCG "3"       "NK_cells"        "9.96125097848532e-18"
## AAACGCACTGGTAC "1"       "T_cells"         "5.18028174072102e-06"
```

Plot the data with cell type

```
library(Seurat)
pbmc <- pbmc_res$obj
pbmc <- Seurat::RunTSNE(pbmc, dims = 1:30)
Seurat::DimPlot(pbmc, reduction = "tsne", label = TRUE, pt.size = 0.5) + NoLegend()
```



For single cell ATAC sequence data

To deal with single cell ATAC sequence data, you may need following codes to install extra dependent packages.

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("GenomeInfoDb")
```

```
BiocManager::install("ensembldb")
BiocManager::install("EnsDb.Hsapiens.v75")
BiocManager::install("GenomicRanges")

library(devtools)
install_github("timoast/signac")
```

Quick example

One can also use our main function "assignCellType" to do analysis in one step. Here we use a toy data set "small_ATAC", which contains 685 peaks and 80 cells. Note that here we set datatype = "ATAC" and "annotation.file" is the path to GTF annotation file.

```
annotation.file <- "~/Documents/Single cell/package example/R package/atac2rna/dataset/Homo_sapiens.GRCh
ATAC_example_res <- assignCellType(small_ATAC, datatype = "ATAC", annotation.file = annotation.file,
                                   min.cells = 1, min.features = 10, nfeatures = 100, npcs = 10,
                                   dims = 1:10, k.param = 5, resolution = 0.75,
                                   min.pct = 0.25, test.use = "MAST", minSize = 5)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 80
## Number of edges: 2318
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.4055
## Number of communities: 3
## Elapsed time: 0 seconds
## Find marker genes for cluster 0
## Find marker genes for cluster 1
## Find marker genes for cluster 2
## Do GSEA for cluster 0
## Do GSEA for cluster 1
## Do GSEA for cluster 2
```

Same as before, it will return a list with 4 slots.

```
ATAC_example_res$Seurat_obj
```

```
## An object of class Seurat
## 803 features across 80 samples within 2 assays
## Active assay: RNA (254 features)
##  1 other assay present: peaks
##  1 dimensional reduction calculated: pca
```

```
head(ATAC_example_res$cell_mat)
```

```
##                     ClusterID Cell Type      padj
## ATTACTCTCGATGCAT-1 "1"       "T_cells"      "0.00122908718329539"
## AGCCCGAGTCTGGTCG-1 "1"       "T_cells"      "0.00122908718329539"
## GCTCAGGTCGCGTGAC-1 "1"       "T_cells"      "0.00122908718329539"
## ATGCATGGTTGAATAG-1 "2"       "unidentified" NA
## TTGAGCACACTGTCAA-1 "2"       "unidentified" NA
## GGGTTATTCGCCTTAC-1 "2"       "unidentified" NA
```

```
head(ATAC_example_res$cluster_celltype)
```

```
##          cell type      NES                padj
## Cluster0 "T_cells"      "2.15365861133432" "0.00122908718329539"
## Cluster1 "unidentified" NA                 NA
## Cluster2 "unidentified" NA                 NA
```

One can also go through the pipeline step by step. Note that the only difference is before step 1 ("scqc" function), we need to use "atac2rna" function to create gene activity matrix. The following steps are same.

```
## don't run
ATAC_example <- atac2rna(small_ATAC, annotation.file = annotation.file)
ATAC_example <- scqc(ATAC_example, min.cells = 1, min.features = 10, nfeatures = 100, npcs = 10)
ATAC_example <- doClustering(ATAC_example, dims = 1:10, k.param = 5, resolution = 0.75)
cluster_list <- getFC(ATAC_example, min.pct = 0.25, test.use = "MAST")
cluster_celltype <- doGSEA(cluster_list = cluster_list, minSize = 5)
ATAC_example_res <- labelCelltype(ATAC_example, cluster_celltype)
```

"atac2rne" function creates gene activity matrix bases on "CreateGeneActivityMatrix" function in "Seurat" package, which is to take in a peak matrix and an annotation file (gtf), and collapse the peak matrix to a gene activity matrix. It makes the simplifying assumption that all counts in the gene body plus X kb up and or downstream should be attributed to that gene. Please try "?atac2rna" to check details for parameters.

For other data type.

One can also go through the pipeline for other data types, for example, "Cell surface proteins". Here we will only focus on the difference.

In the first step, "scqc" function will only create Seurat object and save the matrix. One need to set data type.

```
obj <- scqc(dta, datatype = "Antibody Capture", min.cells = 1, min.features = 10)
```

In clustering, we will do hierarchical clustering for other data types. "hclustmethod" is the agglomeration method to be used for hierarchical clustering and "ncluster" is the number of clusters you want. One need to set data type.

```
obj <- doClustering(obj, datatype = "Antibody Capture")
```

In getting fold change, we suggest using the "Wilcoxon Rank Sum test".

```
cluster_list <- getFC(obj, min.pct = 0.25, test.use = "wilcox")
```

Doing GSEA, one needs to provide a cell type markers database. Setting "db" to be a path to the new database that to be used, the file must be in 'rds' format.

```
cluster_celltype <- doGSEA(cluster_list = cluster_list, db = "path/to/your/rdsfile")
```

The last step, label the cell type, is the same as before.

```
obj_res <- labelCelltype(obj, cluster_celltype)
```