Pseudo-bulk based comparison of cell types in two Seurat objects

Minzhe Guo, Cheng Jiang

Compiled: 2022-07-30

Introduction

In this document, We will be using two publicly-available human PBMC datasets as an example to demonstrate the comparison of cell types in two Seurat objects based on pseudo-bulk expression profiles and highly variable genes.

Loading pacakge

First, load in Seurat and packages that will be used in this demo.

```r
library(Seurat)
library(pheatmap)
library(ggplot2)
library(patchwork)
```

Data preparation

The example datasets that we will use for demonstration are available through SeuratData package (https://github.com/satijalab/seurat-data).

```r
# load the SeuratData pacakge
library(SeuratData)
```

Let's download the two example human PBMC datasets from SeuratData using InstallData().

```r
InstallData("ifnb")
InstallData("pbmcsca")
```

Load the first dataset "ifnb" and assign it to the variable "data1". Check the object and avaialble cell metadata information. We can see that the data is already a Seurat object, containing an RNA assay of 14,503 features in 13,999 cells. The celltype annotations are in the "seurat_annotations" column of meta.data. The data is likely not normalized, as the max value of the "data" slot is a large integer. Run NormalizedData() to normalize the data.

```r
data("ifnb")
data1 = ifnb

data1
```

```
## An object of class Seurat
## 14053 features across 13999 samples within 1 assay
## Active assay: RNA (14053 features, 0 variable features)
```

```
head(data1@meta.data)
```

```
##                     orig.ident nCount_RNA nFeature_RNA stim seurat_annotations
## AAACATACATTTCC.1 IMMUNE_CTRL       3017          877 CTRL          CD14 Mono
## AAACATACCAGAAA.1 IMMUNE_CTRL       2481          713 CTRL          CD14 Mono
## AAACATACCTCGCT.1 IMMUNE_CTRL       3420          850 CTRL          CD14 Mono
## AAACATACCTGGTA.1 IMMUNE_CTRL       3156         1109 CTRL                pDC
## AAACATACGATGAA.1 IMMUNE_CTRL       1868          634 CTRL      CD4 Memory T
## AAACATACGGCATT.1 IMMUNE_CTRL       1581          557 CTRL          CD14 Mono
```

```
max(data1@assays$RNA@data)
```

```
## [1] 3828
```

```
data1 = NormalizeData(data1, verbose = F)
```

Load the second dataset “pbmcsca” and assign it to the variable “data2”. Check the object and avaialble cell metadata information. We can see that the data is also a Seurat object, containing an RNA assay of 33,694 features in 31,021 cells. The celltype annotations are in the “CellType” column of meta.data. The data is also likely not normalized, as the max value of the “data” slot is a large integer. Run NormalizedData() to normalize the data.

```
data("pbmcsca")
data2 = pbmcsca

data2
```

```
## An object of class Seurat
## 33694 features across 31021 samples within 1 assay
## Active assay: RNA (33694 features, 0 variable features)
```

```
head(data2@meta.data)
```

```
##                     orig.ident nCount_RNA nFeature_RNA nGene   nUMI
## pbmc1_SM2_Cell_108      pbmc1     437125         2200  2200 437125
## pbmc1_SM2_Cell_115      pbmc1     335596         2438  2438 335596
## pbmc1_SM2_Cell_133      pbmc1     302204         1874  1874 302204
## pbmc1_SM2_Cell_142      pbmc1     377420         2480  2480 377420
## pbmc1_SM2_Cell_143      pbmc1     385514         2196  2196 385514
## pbmc1_SM2_Cell_144      pbmc1     304994         2216  2216 304994
##                          percent.mito Cluster         CellType Experiment
## pbmc1_SM2_Cell_108 0.0297434465355702       0 Cytotoxic T cell      pbmc1
## pbmc1_SM2_Cell_115 0.0311521658159055       0 Cytotoxic T cell      pbmc1
## pbmc1_SM2_Cell_133 0.0431128105727693       0 Cytotoxic T cell      pbmc1
## pbmc1_SM2_Cell_142 0.0260323569927476       0 Cytotoxic T cell      pbmc1
## pbmc1_SM2_Cell_143 0.0404759383962183       0 Cytotoxic T cell      pbmc1
## pbmc1_SM2_Cell_144  0.023409951391094       0 Cytotoxic T cell      pbmc1
##                          Method
## pbmc1_SM2_Cell_108 Smart-seq2
## pbmc1_SM2_Cell_115 Smart-seq2
## pbmc1_SM2_Cell_133 Smart-seq2
## pbmc1_SM2_Cell_142 Smart-seq2
## pbmc1_SM2_Cell_143 Smart-seq2
## pbmc1_SM2_Cell_144 Smart-seq2
```

```
max(data2@assays$RNA@data)
```

```
## [1] 34710
```

```
data2 = NormalizeData(data2, verbose = F)
```

Create pseudo-bulk profiles

Create a pseudo-bulk profile for each cell type in each dataset. A pseudo-bulk profile is comprised of each gene's averaged expression in a cell type.

```
# set the cell type annotation as the active identity of the Seurat object
data1 = SetIdent(data1, value = data1@meta.data$seurat_annotations)
data2 = SetIdent(data2, value = data2@meta.data$CellType)

# Use AverageExpression() to calculate each gene's average expression in each cell type.  By
# setting return.seurat=T, the results will be returned as a Seurat object
data1.avg = AverageExpression(data1, assay = "RNA", slot = "data", return.seurat = T)
data2.avg = AverageExpression(data2, assay = "RNA", slot = "data", return.seurat = T)
```

Find highly variable genes

Find highly variable genes that have expression in both datasets for the correlation calculation.

```
# first, find top 2000 most highly variable genes (HVGs) within each dataset
data1.avg = FindVariableFeatures(data1.avg, nfeatures = 2000)
data2.avg = FindVariableFeatures(data2.avg, nfeatures = 2000)

# union the HVGs from the two datasets
hvg.use = union(data1.avg@assays$RNA@var.features, data2.avg@assays$RNA@var.features)
cat(length(hvg.use), "HVGs identitied from the two datasets\n")
```

```
## 3127 HVGs identitied from the two datasets
```

```
# keep HVGs that have expression in both datasets
hvg.use = hvg.use[which(hvg.use %in% rownames(data1.avg@assays$RNA@data))]
hvg.use = hvg.use[which(hvg.use %in% rownames(data2.avg@assays$RNA@data))]

cat("Use", length(hvg.use), "HVGs that have expression in both datasets\n")
```

```
## Use 2676 HVGs that have expression in both datasets
```

Scale pseudo-bulk expression of the HVGs

```
data1.avg = ScaleData(data1.avg, features = hvg.use)
data2.avg = ScaleData(data2.avg, features = hvg.use)
```

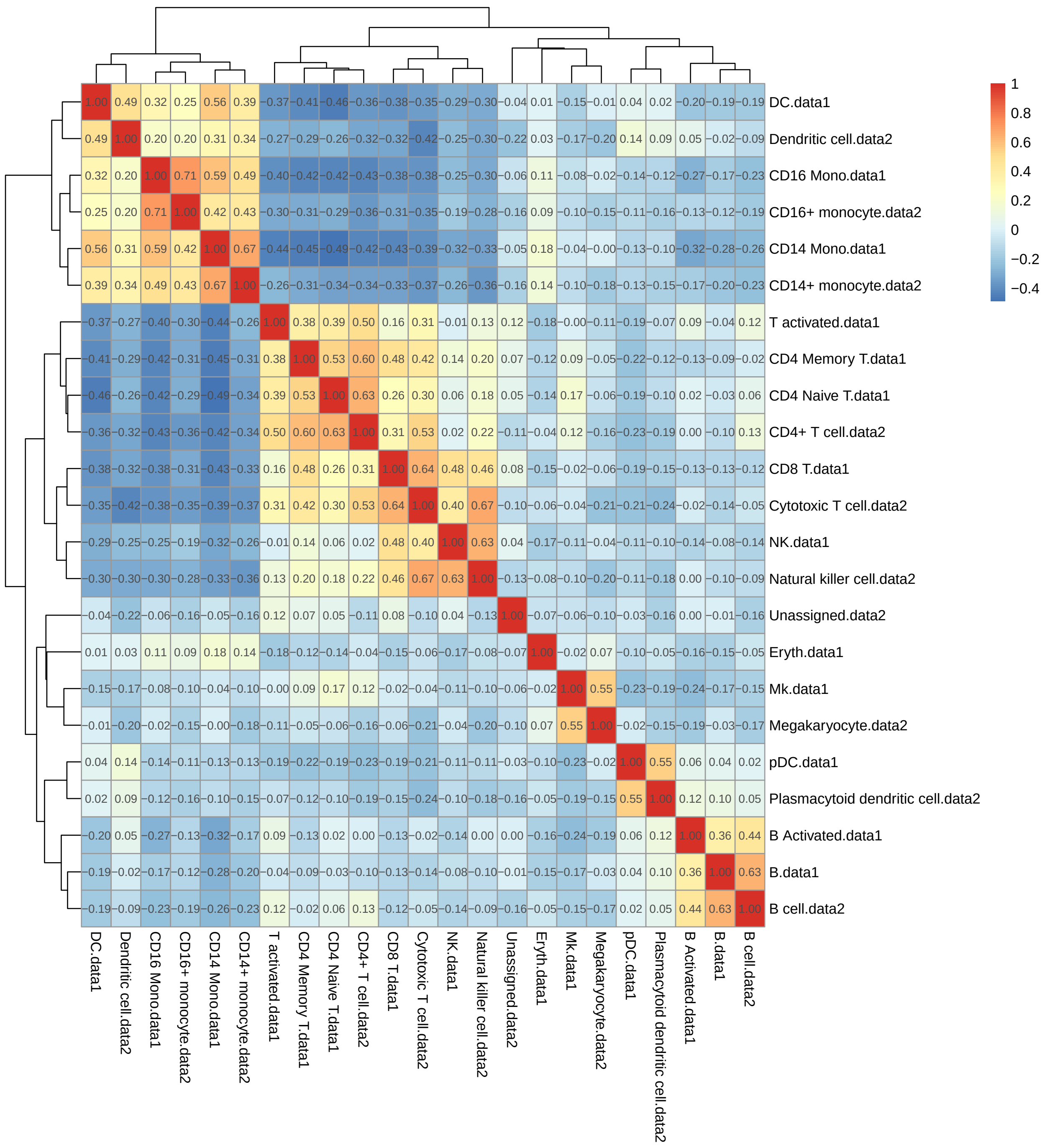
Create a matrix that combines the scaled pseudo-bulk expression of HVGs in different datasets

```
# get the scaled pseudo-bulk expression of the HVGs
data1.data = data1.avg@assays$RNA@scale.data
data2.data = data2.avg@assays$RNA@scale.data

# add suffix to column names (cell types) to distinguish the data from different datasets
colnames(data1.data) = paste0(colnames(data1.data), ".data1")
colnames(data2.data) = paste0(colnames(data2.data), ".data2")

# combine the data to create an expression matrix
expr_mat = cbind(data1.data, data2.data[rownames(data1.data), ])

# make sure no missing values generated
any(is.na(expr_mat))
```

```
## [1] FALSE
```

Correlations and Visualization

```
# calcualte the correlations of cell types using scaled pseudo-bulk expression of HVGs
cor_mat = cor(expr_mat)

# visualize the correlations using pheatmap; by setting display_numbers = T, the correlation
# values will be shown on the heatmap; use the round() function to round the correlation
# values to two decimal places for better display
g = pheatmap::pheatmap(round(cor_mat, digits = 2), clustering_method = "complete", display_numbers = T)
```



```
# save the heatmap to a tif file
tiff(filename = "cor_heatmap.tif", width = 12, height = 11, res = 300, units = "in", compression = "lzw
print(g)
dev.off()
```

```
## pdf
##   2
```

```r
# save the HVG expression matrix and the correlation matrix as source data
sourcedata = list(hvg.expression = expr_mat, correlations = cor_mat)
save(sourcedata, file = "sourcedata.rda")
```

Session Info

```r
sessionInfo()
```

```
## R version 4.1.0 (2021-05-18)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19042)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] pbmcsca.SeuratData_3.0.0      pbmcMultiome.SeuratData_0.1.3
##  [3] pancreasref.SeuratData_1.0.0  panc8.SeuratData_3.0.2
##  [5] ifnb.SeuratData_3.1.0         SeuratData_0.2.1
##  [7] patchwork_1.1.1               ggplot2_3.3.5
##  [9] pheatmap_1.0.12               SeuratObject_4.0.4
## [11] Seurat_4.1.0
##
## loaded via a namespace (and not attached):
##   [1] Rtsne_0.15            colorspace_2.0-3      deldir_1.0-6
##   [4] ellipsis_0.3.2        ggridges_0.5.3        rstudioapi_0.13
##   [7] spatstat.data_2.1-2   leiden_0.3.9          listenv_0.8.0
##  [10] ggrepel_0.9.1         fansi_1.0.2           codetools_0.2-18
##  [13] splines_4.1.0         knitr_1.37            polyclip_1.10-0
##  [16] jsonlite_1.7.3        ica_1.0-2             cluster_2.1.2
##  [19] png_0.1-7             uwot_0.1.11           shiny_1.7.1
##  [22] sctransform_0.3.3     spatstat.sparse_2.1-0 compiler_4.1.0
##  [25] httr_1.4.2            assertthat_0.2.1      Matrix_1.4-0
##  [28] fastmap_1.1.0         lazyeval_0.2.2        cli_3.1.1
##  [31] later_1.3.0           formatR_1.11          htmltools_0.5.2
##  [34] tools_4.1.0           igraph_1.2.11         gtable_0.3.0
##  [37] glue_1.6.1            RANN_2.6.1            reshape2_1.4.4
##  [40] dplyr_1.0.7           rappdirs_0.3.3        Rcpp_1.0.8
##  [43] scattermore_0.7       vctrs_0.3.8           nlme_3.1-155
##  [46] lmtest_0.9-39         xfun_0.29             stringr_1.4.0
##  [49] globals_0.14.0        mime_0.12             miniUI_0.1.1.1
##  [52] lifecycle_1.0.1       irlba_2.3.5           goftest_1.2-3
##  [55] future_1.23.0         MASS_7.3-55           zoo_1.8-9
##  [58] scales_1.1.1          spatstat.core_2.3-2   promises_1.2.0.1
##  [61] spatstat.utils_2.3-0  parallel_4.1.0        RColorBrewer_1.1-2
##  [64] yaml_2.2.2            reticulate_1.24       pbapply_1.5-0
##  [67] gridExtra_2.3         rpart_4.1.16          stringi_1.7.6
##  [70] highr_0.9             rlang_1.0.1           pkgconfig_2.0.3
##  [73] matrixStats_0.61.0    evaluate_0.14         lattice_0.20-45
##  [76] ROCR_1.0-11           purrr_0.3.4           tensor_1.5
##  [79] htmlwidgets_1.5.4     cowplot_1.1.1         tidyselect_1.1.1
##  [82] parallelly_1.30.0     RcppAnnoy_0.0.19      plyr_1.8.6
##  [85] magrittr_2.0.2        R6_2.5.1              generics_0.1.2
##  [88] DBI_1.1.2             withr_2.5.0           pillar_1.7.0
##  [91] mgcv_1.8-38           fitdistrplus_1.1-6    survival_3.2-13
##  [94] abind_1.4-5           tibble_3.1.6          future.apply_1.8.1
##  [97] crayon_1.5.0          KernSmooth_2.23-20    utf8_1.2.2
## [100] spatstat.geom_2.3-1   plotly_4.10.0         rmarkdown_2.13
## [103] grid_4.1.0            data.table_1.14.2     digest_0.6.29
## [106] xtable_1.8-4          tidyr_1.2.0           httpuv_1.6.5
## [109] munsell_0.5.0         viridisLite_0.4.0
```