01) Standard Seurat analysis up to DEG identification

Kane Toh*

2023-12-18

Contents

1 Loading libraries

```r
library(Seurat)  # For scRNAseq analysis
library(dplyr)   # For data cleaning/wrangling
library(tidyr)   # For extra data cleaning tasks
library(stringr) # string manipulation
library(biomaRt) # To convert ensembl_ids to gene symbols
library(ggplot2) # For plotting
library(cowplot) #For placing plots in a grid, similar to patchwork
library(ArchR) #Library for scATAC-seq analysis, but with awesome color palettes
```

*Genomics and Data Analytics Core (GeDaC), Cancer Science Institute of Singapore, National University of Singapore; kanetoh@nus.edu.sg

```
# Loads in custom functions that I've written to assist with plotting/saving tasks
source("helper.R")

# Path to saving the results fo the analysis
res_path <- "../results/02_standardSeurat/"
if (!dir.exists(res_path)){

  dir.create(res_path)
}else{
  print("Results directory exists")
}
```

```
## [1] "Results directory exists"
```

```
# Data path specifes where all the outputs from the parseBio split-pipe pipeline is stored.
data_path <- "../data/"
if (!dir.exists(data_path)){
  warning("Directory does not exist: ", data_path)
}else{
  print("Data directory exists")
}
```

```
## [1] "Data directory exists"
```

```
# RDS path specifies where all the intermediate generated R objects will be saved.
rds_path <- "../rds/"
if (!dir.exists(rds_path)){
  warning("Directory does not exist: ", rds_path)
}else{
  print("RDS directory exists")
}
```

```
## [1] "RDS directory exists"
```

2 Loading in QCed seurat object

```
mbladder <- ReadObject(rds_path, "01_seuratMerged_postQC")
```

3 Add useful metadata column to identify samples

```
# This creates a new column in the seurat metadata slot containing the sample name
mbladder@meta.data <- mbladder@meta.data %>%
  tidyr::unite("sample_name", Condition:Type, sep="_", remove=FALSE)
```

4 Perform Standard Seurat QC steps.

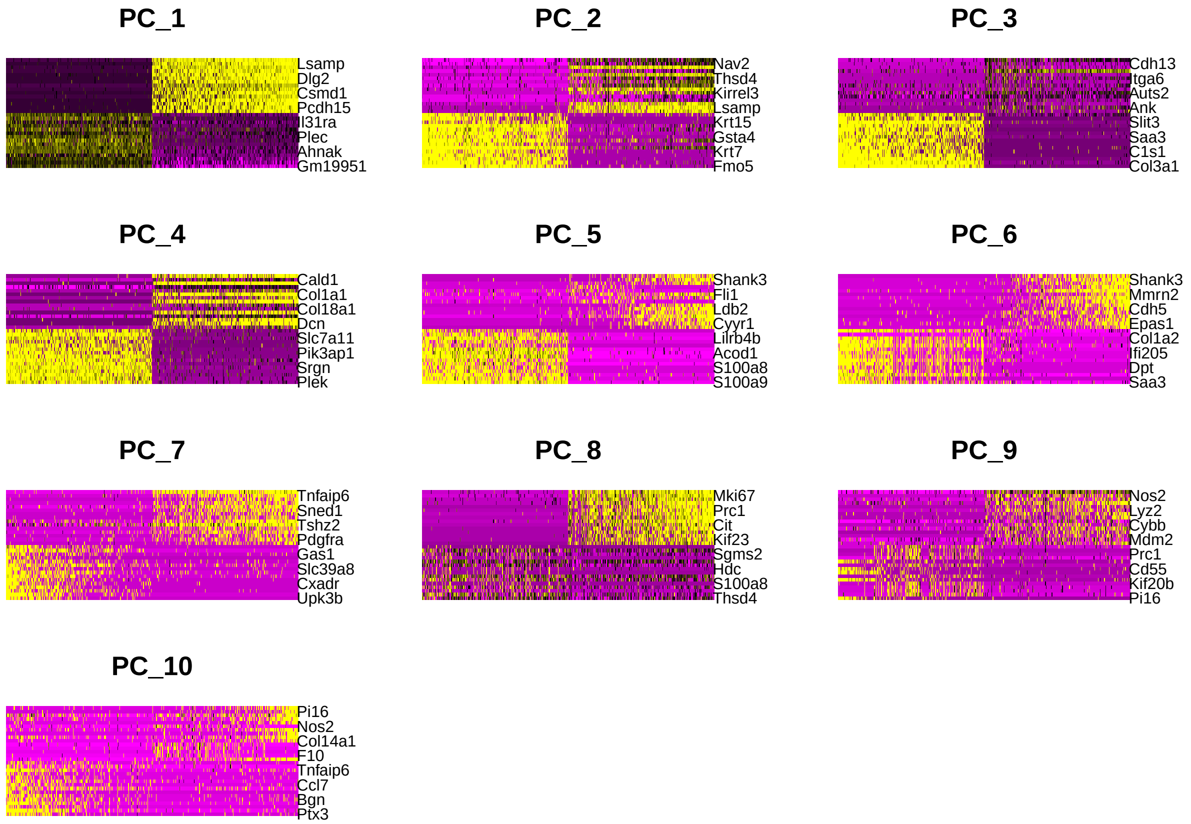
This includes log-normalising the data; finding variable features; scaling for visualisation, identifying top few PCs to use for UMAP visualisation, as well as finding nearest neighbors in a SNN and then using standard louvain community detection for cluster identification.

```
mbladder <- NormalizeData(
  object = mbladder,
  normalization.method = "LogNormalize",
  scale.factor = 10000)
mbladder <- FindVariableFeatures(mbladder)
mbladder <- ScaleData(mbladder)
#seurat_harmony@assays$RNA@scale.data
# Perform PCA and color by cell cycle phase
mbladder <- RunPCA(mbladder)
mbladder <- RunUMAP(mbladder, reduction = "pca", dims = 1:15)
mbladder <- FindNeighbors(mbladder, reduction = "pca", dims = 1:15)
mbladder <- FindClusters(mbladder, resolution = 0.5)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 10016
## Number of edges: 362690
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.9007
## Number of communities: 13
## Elapsed time: 1 seconds
```

Viewing PCA related statistics such as elbow-plot to identify top few PCs, as well as heatmap showing the top PCs.

```
eb <- ElbowPlot(mbladder) # Around 10-15 pcs enough.
#SaveFigure(eb, "elbowplot_unintegrated", width = 12, height = 6)
pca_heatmap <- DimHeatmap(mbladder,dims=1:10, cells=500)
```



```r
#SaveFigure(pca_heatmap, "pcaheatmap_unintegrated", width = 12, height = 20)
```

5 Identifying cell cycle

We observe that cells do not separate by cell-cycle phase, so we do not regress out the cell-cycle.

```r
url<-"https://raw.githubusercontent.com/hbc/tinyatlas/master/cell_cycle/Mus_musculus.csv"
cc_file <- RCurl::getURL(url)
cell_cycle_genes <- read.csv(text = cc_file)

mart <- useMart("ENSEMBL_MART_ENSEMBL",verbose = TRUE)
```

```
## BioMartServer running BioMart version: 0.7
## Mart virtual schema: default
## Mart host: https://www.ensembl.org:443/biomart/martservice
```

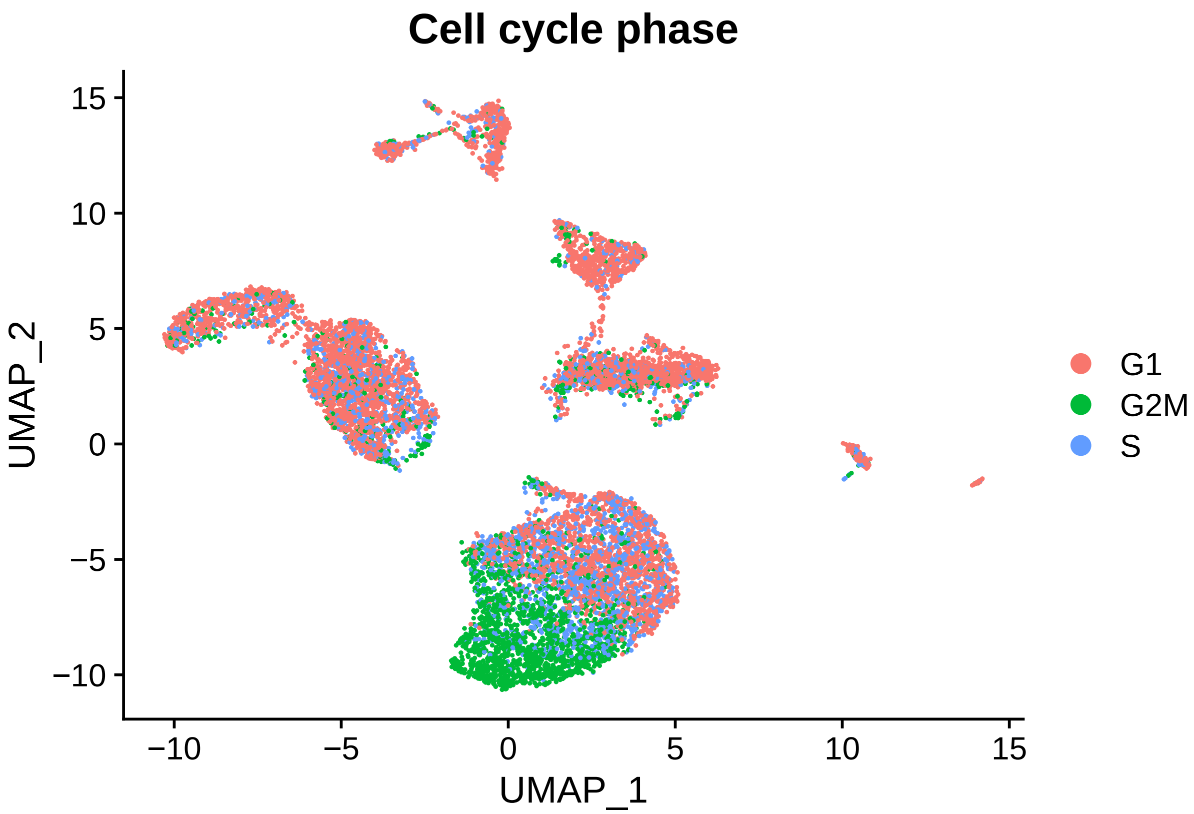
```r
mart <- useDataset("mmusculus_gene_ensembl", mart)

annotLookup <- getBM(
  mart=mart,
  attributes=c("ensembl_transcript_id", "ensembl_gene_id",
               "gene_biotype", "external_gene_name"),
  filter="ensembl_gene_id",
  values=cell_cycle_genes$geneID,
  uniqueRows=TRUE)

cell_cycle_markers <- dplyr::left_join(cell_cycle_genes, annotLookup, by = c("geneID" = "ensembl_gene_id

# Acquire the S phase genes
s_genes <- cell_cycle_markers %>%
        dplyr::filter(phase == "S") %>%
        pull("external_gene_name") %>%
        unique()
# Acquire the G2M phase genes
g2m_genes <- cell_cycle_markers %>%
        dplyr::filter(phase == "G2/M") %>%
        pull("external_gene_name") %>%
        unique()

# Perform cell cycle scoring
mbladder <- CellCycleScoring(mbladder,
                                   g2m.features = g2m_genes,
                                   s.features = s_genes,
                             set.ident=TRUE)
# Visualize the PCA, grouping by cell cycle phase
cc_dimplot <- DimPlot(mbladder,
        reduction = "umap",
        group.by= "Phase") +
  ggtitle("Cell cycle phase")
cc_dimplot
```



```
#SaveFigure(cc_dimplot, "cc_dimplot", width = 8, height = 6)

# Resetting the identity of each cell to the seurat clusters after cell-cycle phase analysis
Idents(mbladder) <- "seurat_clusters"
```

6 Viewing UMAP

6.1 View seurat cluster labels

```
cell_types_colors <- ArchR::paletteDiscrete(mbladder@meta.data$seurat_clusters, set="stallion")
umap_seurat <- DimPlot(mbladder, reduction="umap", cols =
          cell_types_colors)
```

6.2 View by sample_name and cell cycle phase

```
umap_seurat_samplenames <- DimPlot(mbladder, reduction="umap", split.by="sample_name", cols = cell_types
  theme(legend.position="None")

umap_seurat_cellCycle <- DimPlot(mbladder, reduction="umap", split.by="Phase", cols=
          cell_types_colors)+
          theme(legend.position="None")
```

6.3 Combined plots

```
top_row <- cowplot::plot_grid(umap_seurat, nrow = 1, labels = c("A"), label_size = 18)
bottom_row <- cowplot::plot_grid(umap_seurat_samplenames, umap_seurat_cellCycle,  nrow = 1, labels=c("B
combined_umaps <- cowplot::plot_grid(top_row, bottom_row, nrow=2)

SaveFigure(res_path = res_path,
           plots = combined_umaps,
           name = "combined_umaps_unannotated",
           width = 15,
           height = 8)
```

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