lung_reference

2025-07-18

R Markdown

This script prepares a publicly-available single cell RNA sequencing data set, annotated by the cell type, for processing as a reference for clustering using Seurat.

Citation:

Zepp, J. A., [. . .], Morrisey, E. E. (2021). Genomic, epigenomic, and biophysical cues controlling the emergence of the lung alveolus. Science (American Association for the Advancement of Science), 371(6534). https://doi.org/10.1126/science.abc3172

Click here to download data

```r
library(anndata)
library(Seurat)
library(MuDataSeurat)
library(BPCells)
library(reticulate)
library(Seurat)
library(biomaRt)
library(tidyverse)
library(ggpubr)
```

Read data

```r
path <- "./data/mouse_atlas/cellXgene.h5ad"
# h5ad file was formatted in Python; convert to a format available to R
numpy <- import('numpy', convert = FALSE)
anndata <- import('anndata', convert = FALSE)
scipy <- import('scipy', convert = FALSE)

adata <- py_to_r(read_h5ad(path))
mat <- as.sparse(adata$X)
# Transpose
tmat <- t(mat)
```

At this point, row names are Ensembl IDs.

To match the intended Seurat object, gene names are needed.

This can be matched using BioMart

```r
mart <- useDataset("mmusculus_gene_ensembl", useMart("ensembl"))
gIDs <- row.names(tmat)
gNames <- getBM(filters = "ensembl_gene_id",
                attributes = c(
                  "ensembl_gene_id",
                  "external_gene_name"),
                values = gIDs,
                mart = mart)
```

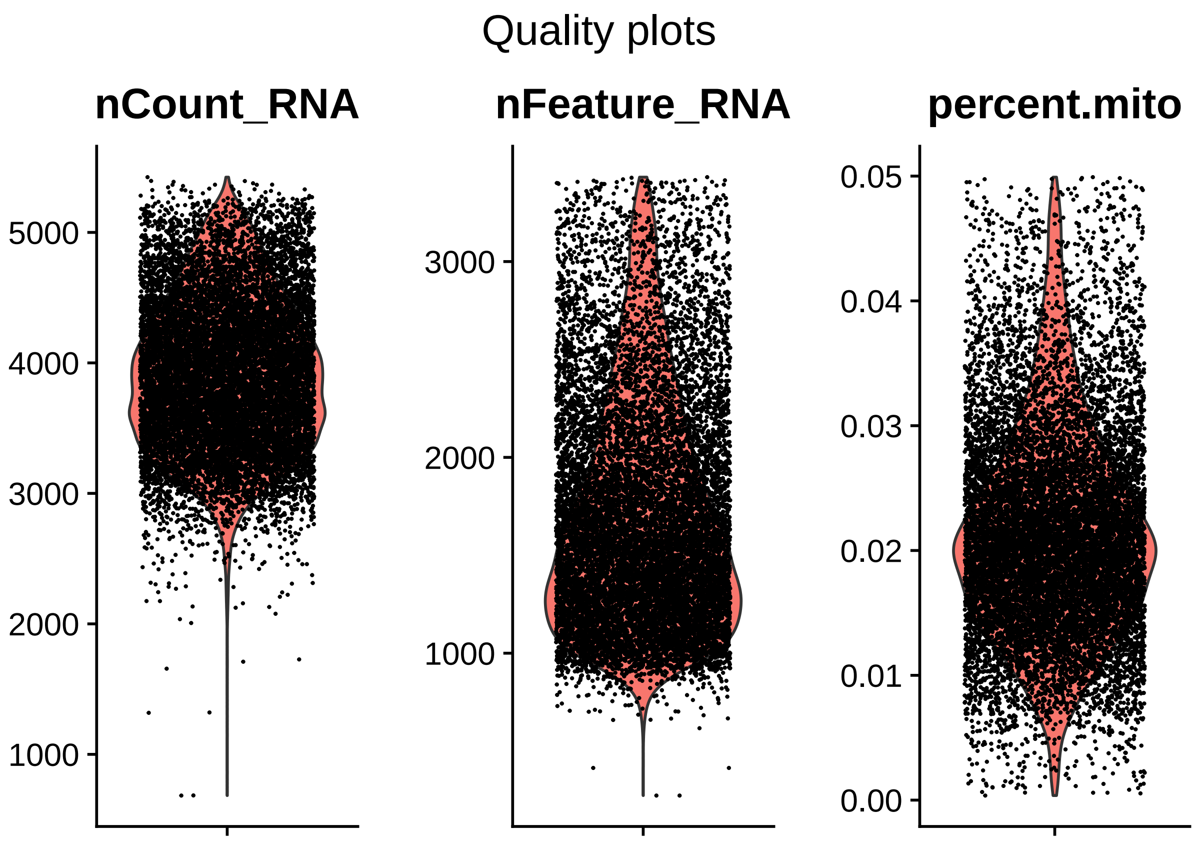
```r
# Some IDs returned results, but no gene names
# In order to use as row names, empty rows will not be compatible
# Fill those in with the ensembl IDs
for (i in 1:nrow(gNames)) {
  if (gNames$external_gene_name[i] == "") {
    gNames$external_gene_name[i] <- gNames$ensembl_gene_id[i]
  }
}
# Some IDs returned nothing and those also need to be added in
# Create a dummy data set with those IDs and add that in
dummy <- cbind(gIDs[which(!(rownames(tmat) %in% gNames$ensembl_gene_id))],
               gIDs[which(!(rownames(tmat) %in% gNames$ensembl_gene_id))])
colnames(dummy) <- colnames(gNames)
gNames <- rbind(gNames, dummy)
# Gene names df needs to match order of the matrix to correctly use as row names
gNames <- gNames[match(rownames(tmat), gNames$ensembl_gene_id),]
# Final check before replacing row names
if (identical(gNames$ensembl_gene_id, rownames(tmat)) == FALSE) {
  stop()
}
rownames(tmat) <- make.unique(gNames$external_gene_name)
```

Create the Seurat object

```r
lung <- CreateSeuratObject(counts = tmat,
                           meta.data = adata$obs )
# Intended use is for comparison to adult
# Dropping other developmental stages from the reference will speed processing time
lung <- subset(lung, development_stage == "prime adult stage")
```

Look at the QC metrics

```r
plots <- c()
feats <- c("nCount_RNA", "nFeature_RNA", "percent.mito")
# Plot each feature separately
for (i in 1:length(feats)) {
  temp <- VlnPlot(lung,
                  features = feats[i]) +
    theme(axis.text.x = element_blank(),
          axis.title.x = element_blank(),
          legend.position = "none")
  plots <- c(plots, list(temp))
}
# Generate plot
ggarrange(plotlist = plots, nrow = 1) %>%
  annotate_figure(top = text_grob("Quality plots",
                                  size = 16))
```



The data quality looks good and was very likely pre-filtered before upload.

No additional filtering necessary.

Normalize and scale

```r
lung <- NormalizeData(lung)
```

```
## Normalizing layer: counts
```

```r
all.genes <- rownames(lung)
lung <- ScaleData(lung, features = all.genes)
```

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

Cluster

```r
lung <- FindVariableFeatures(lung)
```

```
## Finding variable features for layer counts
```

```r
lung <- RunPCA(lung,
                   features = VariableFeatures(object = lung),
                verbose = FALSE)

lung <- FindNeighbors(lung, dims = 1:16)
```

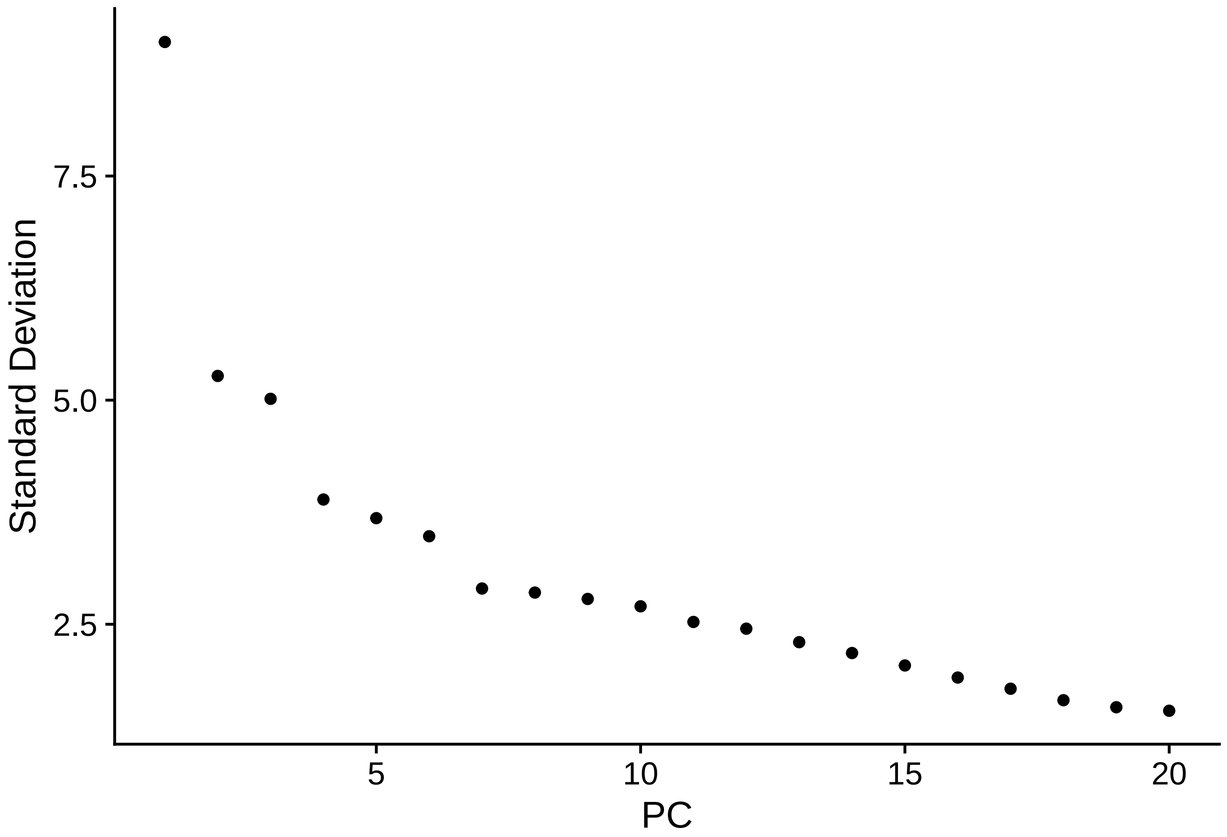
```
## Computing nearest neighbor graph
```

```
## Computing SNN
```

```r
lung <- FindClusters(lung, resolution = 0.8)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 12213
## Number of edges: 454300
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8875
## Number of communities: 24
## Elapsed time: 1 seconds
```

```
ElbowPlot(lung)
```

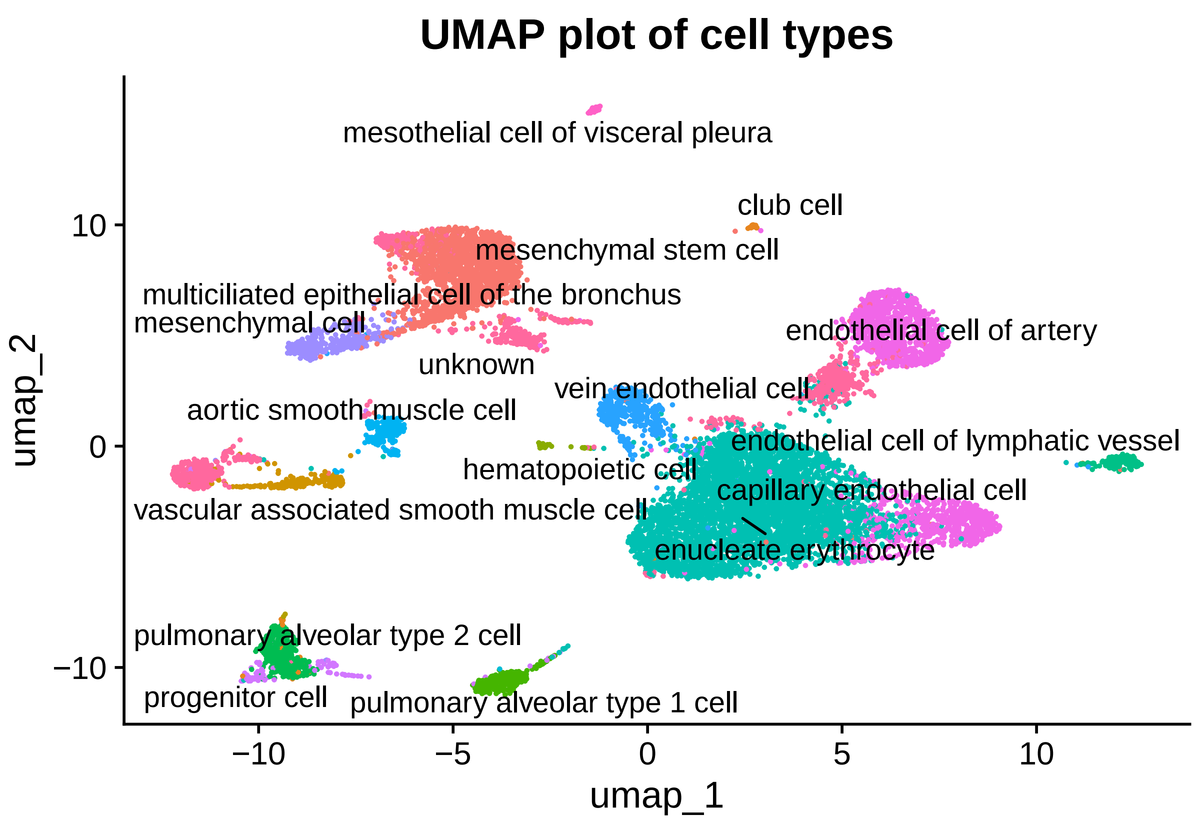


The "elbow," point of diminishing returns, happens between the component ~15 and 20. (An ideal elbow plot would have a harder, more defined elbow.) I will use 16.

Dimensional reduction

```
lung <- SCTransform(lung, verbose = FALSE)

lung <- RunUMAP(lung, dims = 1:16, verbose = FALSE)
lung <- RunTSNE(lung, dims = 1:16, verbose = FALSE)
DimPlot(lung,
        reduction = "umap",
        repel = TRUE,
        label = TRUE,
        group.by = "cell_type",
        ) +
  NoLegend() +
  ggtitle("UMAP plot of cell types")
```



Export data to be used by other scripts

```
saveRDS(lung, "./data/mouse_atlas/lung_reference.rds")
```

Darwin, 24.5.0, Darwin Kernel Version 24.5.0: Tue Apr 22 19:54:26 PDT 2025; root:xnu-11417.121.6~2/RELEASE_ARM64_T8
Emilys-MacBook-Air.local, arm64, root, emilykibbler, emilykibbler