```
> # Chunk 1: setup
> knitr::opts_chunk$set(
+   cache = FALSE,
+   cache.lazy = FALSE,
+   tidy = TRUE
+ )
>
> # Chunk 2: init
> #load all the librarys needed and set path by pressing "session"" -> "set working
directory" -> "to source file location"
>
> library(SingleR)
> library(Seurat)
> library(dplyr)
> library(devtools)
> ctrl_samp.name = "CP1_dge_s"
> stim_samp.name = "CS1_dge_s"
>
> # Chunk 3: createseurat
> library(Seurat)
> ########################## Set up control object
> ctrl.data <- read.table(file = paste0(ctrl_samp.name, ".txt.gz"), header = TRUE, row.names
= 1, colClasses =c("character", rep("numeric", 1000)))
> #ctrl.data <- read.table(file = paste0(ctrl_samp.name, "_dge.txt.gz"), header = TRUE,
row.names = 1, colClasses =c("character", rep("numeric", 10000)))
>
> # Create seurat object
> ctrl <- CreateSeuratObject(raw.data = ctrl.data, project = "IMMUNE_CTRL", min.cells = 1,
min.genes = 500)
> ctrl@meta.data$stim <- "CTRL"
> colnames(ctrl.data)<- paste0(colnames(ctrl.data),"_CTRL")
>
> # mito genes to meta data
> mito.genes <- grep(pattern = "^mt-", x = rownames(x = ctrl@data), value = TRUE)
> percent.mito <- Matrix::colSums(ctrl@raw.data[mito.genes, ]) /
Matrix::colSums(ctrl@raw.data)
>
> # AddMetaData adds columns to object@meta.data, and is a great place to stash QC stats
> ctrl <- AddMetaData(object = ctrl, metadata = percent.mito, col.name = "percent.mito")
>
> #visualize for choosing thresholds
> VlnPlot(object = ctrl, features.plot = c("nGene", "nUMI", "percent.mito"), nCol = 3)
>
> ########################## Set up stimulated object
>
> stim.data <- read.table(file = paste0(stim_samp.name, ".txt.gz"), header = TRUE, row.names
= 1, colClasses =c("character", rep("numeric", 1000)))
> #stim.data <- read.table(file = paste0(stim_samp.name, "_dge.txt.gz"), header = TRUE,
row.names = 1, colClasses =c("character", rep("numeric", 10000)))
> colnames(stim.data)<- paste0(colnames(stim.data),"_STIM")
>
>
> #Create Seurat object
> stim <- CreateSeuratObject(raw.data = stim.data, project = "IMMUNE_STIM", min.cells = 1,
min.genes = 500)
> stim@meta.data$stim <- "STIM"
>
> # mito genes to meta data
> mito.genes <- grep(pattern = "^mt-", x = rownames(x = stim@data), value = TRUE)
> percent.mito <- Matrix::colSums(stim@raw.data[mito.genes, ]) /
Matrix::colSums(stim@raw.data)
>
> # AddMetaData adds columns to object@meta.data, and is a great place to stash QC stats
> stim <- AddMetaData(object = stim, metadata = percent.mito, col.name = "percent.mito")
>
> #visualize for choosing thresholds
> VlnPlot(object = stim, features.plot = c("nGene", "nUMI", "percent.mito"), nCol = 3)
>
> # Chunk 4: thresh
```

```
> ctrl <- FilterCells(object = ctrl, subset.names = c("nGene", "nUMI", "percent.mito"),
low.thresholds = c(200,500, -Inf), high.thresholds = c(1400,2500, 0.02))
> ctrl <- NormalizeData(ctrl)
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> ctrl <- ScaleData(ctrl, display.progress = F)
>
> stim <- FilterCells(object = stim, subset.names = c("nGene", "nUMI", "percent.mito"),
low.thresholds = c(200,500, -Inf), high.thresholds = c(1400,2500, 0.017))
> stim <- NormalizeData(stim)
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> stim <- ScaleData(stim, display.progress = F)
>
>
> # Gene selection for input to CCA
> ctrl <- FindVariableGenes(ctrl, do.plot = F)
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> stim <- FindVariableGenes(stim, do.plot = F)
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> g.1 <- head(rownames(ctrl@hvg.info), 1000)
> g.2 <- head(rownames(stim@hvg.info), 1000)
> genes.use <- unique(c(g.1, g.2))
> genes.use <- intersect(genes.use, rownames(ctrl@scale.data))
> genes.use <- intersect(genes.use, rownames(stim@scale.data))
>
> # Chunk 5: original identity
>
> write_ctrl=rep("ctrl", length(colnames(ctrl@data)))
> write.table(write_ctrl, file="ctrl_orig.ident.txt", sep="\t", eol ="\n",
row.names=colnames(ctrl@data))
>
> write_stim=rep("stim", length(colnames(stim@data)))
> write.table(write_stim, file="stim_orig.ident.txt", sep="\t", eol ="\n",
row.names=colnames(stim@data))
>
> # Chunk 6: singler ctrl
>
> ctrl_raw <- as.matrix(x=ctrl@data)
> singler_ctrl <- CreateSinglerSeuratObject(counts = ctrl_raw, annot="ctrl_orig.ident.txt",
project.name = ctrl_samp.name, min.genes = 500, min.cells = 1, technology = "Microwell-seq",
npca = 2, species = "Mouse", fine.tune = T)
[1] "CP1_dge_s"
[1] "Reading single-cell data..."
[1] "Create Seurat object..."
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
```

```
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Regressing out: nUMI
  |
================================================================================================
==========================================| 100%
Time Elapsed:  11.1471252441406 secs
Scaling data matrix
  |
================================================================================================
==========================================| 100%
[1] "Creat SingleR object..."
[1] "Dimensions of counts data: 15232x708"
[1] "Annotating data with Immgen..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2751"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 03:53
[1] "Number of DE genes:2751"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Immgen (Main types)..."
[1] "Number of DE genes:1814"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:36
[1] "Number of DE genes:1814"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Mouse-RNAseq..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2826"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:36
[1] "Number of DE genes:2826"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Mouse-RNAseq (Main types)..."
[1] "Number of DE genes:2179"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:33
[1] "Number of DE genes:2179"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
```

```
  |
================================================================================================
============================| 100%, Elapsed 00:01
> singler_ctrl.new = convertSingleR2Browser(singler_ctrl)
> saveRDS(singler_ctrl.new, paste0(ctrl_samp.name, "_S4_ctrl.rds"))
> saveRDS(singler_ctrl, paste0(ctrl_samp.name, "_singleR_ctrl.rds"))
>
> # Chunk 7: singler stim
>
> stim_raw <- as.matrix(x=stim@data)
> singler_stim <- CreateSinglerSeuratObject(counts = stim_raw, annot= "stim_orig.ident.txt",
project.name = stim_samp.name, min.genes = 500, min.cells = 1, technology = "Microwell-seq",
species = "Mouse", npca = 10, fine.tune = T)
[1] "CS1_dge_s"
[1] "Reading single-cell data..."
[1] "Create Seurat object..."
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Regressing out: nUMI
  |
================================================================================================
==========================================| 100%
Time Elapsed:  11.7243218421936 secs
Scaling data matrix
  |
================================================================================================
==========================================| 100%
[1] "Creat SingleR object..."
[1] "Dimensions of counts data: 15162x756"
[1] "Annotating data with Immgen..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2739"
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 03:15
[1] "Number of DE genes:2739"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Immgen (Main types)..."
[1] "Number of DE genes:1826"
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:37
[1] "Number of DE genes:1826"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Mouse-RNAseq..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2825"
```

```
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:33
[1] "Number of DE genes:2825"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:00
[1] "Annotating data with Mouse-RNAseq (Main types)..."
[1] "Number of DE genes:2186"
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:30
[1] "Number of DE genes:2186"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:00
> singler_stim.new = convertSingleR2Browser(singler_stim)
> saveRDS(singler_stim.new, paste0(stim_samp.name, "_S4_stim.rds"))
> saveRDS(singler_stim, paste0(stim_samp.name, "_singleR_stim.rds"))
> list_samples<- list(singler_ctrl, singler_stim)
> #list_xy <-
list(singler_ctrl[["seurat"]]@dr[["tsne"]]@cell.embeddings,singler_stim[["seurat"]]@dr[["tsn
e"]]@cell.embeddings)
>
> singler=SingleR.Combine(list_samples)
> saveRDS(singler, paste0(ctrl_samp.name,'_',stim_samp.name,"_tempfinal.rds"))
> singler_ctrl.stim.new = convertSingleR2Browser(singler)
Error in `row.names<-.data.frame`(`*tmp*`, value = value) :
  invalid 'row.names' length
> saveRDS(singler_ctrl.stim.new, paste0(ctrl_samp.name,'_',stim_samp.name, '_final.rds'))
Error in saveRDS(singler_ctrl.stim.new, paste0(ctrl_samp.name, "_", stim_samp.name,  :
  object 'singler_ctrl.stim.new' not found
> View(singler_stim)
> ?CreateSinglerSeuratObject
> ?CreateSinglerObject
> install.packages("/home/drugfarm/proj/tools/SingleR-master.zip")
Installing package into ‘/home/drugfarm/R/x86_64-pc-linux-gnu-library/3.4’
(as ‘lib’ is unspecified)
Warning in install.packages :
  package ‘/home/drugfarm/proj/tools/SingleR-master.zip’ is not available (for R version
3.4.4)
> install.packages("~/proj/tools/SingleR-master.zip", repos = NULL, type = "win.binary")
Installing package into ‘/home/drugfarm/R/x86_64-pc-linux-gnu-library/3.4’
(as ‘lib’ is unspecified)
Error in install.packages : cannot install Windows binary packages on this platform
> install.packages("~/proj/tools/SingleR-master.zip", repos = NULL)
Installing package into ‘/home/drugfarm/R/x86_64-pc-linux-gnu-library/3.4’
(as ‘lib’ is unspecified)
Error in rawToChar(block[seq_len(ns)]) :
  embedded nul in string: 'PK\003\004\n\0\0\0\0\0;GMN\0\0\0\0\0\0\0\0\0\0\0\0\017\0\t\
0SingleR-master/UT\005\0\001\x93Ld\\PK\003\004\n\0\0\0\b\0;GMN\aD\xb9\x93\xf6\001\0\0c\
003\0\0\032\0\t\0SingleR-master/D'
Warning in install.packages :
  installation of package ‘/home/drugfarm/proj/tools/SingleR-master.zip’ had non-zero exit
status
> install.packages("~/proj/tools/SingleR-master.zip", repos = NULL, type = "source")
Installing package into ‘/home/drugfarm/R/x86_64-pc-linux-gnu-library/3.4’
(as ‘lib’ is unspecified)
Error in rawToChar(block[seq_len(ns)]) :
```

```
  embedded nul in string: 'PK\003\004\n\0\0\0\0\0;GMN\0\0\0\0\0\0\0\0\0\0\0\017\0\t\
0SingleR-master/UT\005\0\001\x93Ld\\PK\003\004\n\0\0\0\b\0;GMN\aD\xb9\x93\xf6\001\0\0c\
003\0\0\032\0\t\0SingleR-master/D'
Warning in install.packages :
  installation of package ‘/home/drugfarm/proj/tools/SingleR-master.zip’ had non-zero exit
status
> library(SingleR)
> list_samples<- list(singler_ctrl, singler_stim)
> #list_xy <-
list(singler_ctrl[["seurat"]]@dr[["tsne"]]@cell.embeddings,singler_stim[["seurat"]]@dr[["tsn
e"]]@cell.embeddings)
>
> singler=SingleR.Combine(list_samples)
> saveRDS(singler, paste0(ctrl_samp.name,'_',stim_samp.name,"_tempfinal.rds"))
> singler_ctrl.stim.new = convertSingleR2Browser(singler)
Error in `row.names<-.data.frame`(`*tmp*`, value = value) :
  invalid 'row.names' length
> saveRDS(singler_ctrl.stim.new, paste0(ctrl_samp.name,'_',stim_samp.name, '_final.rds'))
Error in saveRDS(singler_ctrl.stim.new, paste0(ctrl_samp.name, "_", stim_samp.name,  :
  object 'singler_ctrl.stim.new' not found
> # Chunk 1: setup
> knitr::opts_chunk$set(
+   cache = FALSE,
+   cache.lazy = FALSE,
+   tidy = TRUE
+ )
>
> # Chunk 2: init
> #load all the librarys needed and set path by pressing "session"" -> "set working
directory" -> "to source file location"
>
> library(SingleR)
> library(Seurat)
> library(dplyr)
> library(devtools)
> ctrl_samp.name = "CP1_dge_s"
> stim_samp.name = "CS1_dge_s"
>
> # Chunk 3: createseurat
> library(Seurat)
> ############################ Set up control object
> ctrl.data <- read.table(file = paste0(ctrl_samp.name, ".txt.gz"), header = TRUE, row.names
= 1, colClasses =c("character", rep("numeric", 1000)))
> #ctrl.data <- read.table(file = paste0(ctrl_samp.name, "_dge.txt.gz"), header = TRUE,
row.names = 1, colClasses =c("character", rep("numeric", 10000)))
>
> # Create seurat object
> ctrl <- CreateSeuratObject(raw.data = ctrl.data, project = "IMMUNE_CTRL", min.cells = 1,
min.genes = 500)
> ctrl@meta.data$stim <- "CTRL"
> colnames(ctrl.data)<- paste0(colnames(ctrl.data),"_CTRL")
>
> # mito genes to meta data
> mito.genes <- grep(pattern = "^mt-", x = rownames(x = ctrl@data), value = TRUE)
> percent.mito <- Matrix::colSums(ctrl@raw.data[mito.genes, ]) /
Matrix::colSums(ctrl@raw.data)
>
> # AddMetaData adds columns to object@meta.data, and is a great place to stash QC stats
> ctrl <- AddMetaData(object = ctrl, metadata = percent.mito, col.name = "percent.mito")
>
> #visualize for choosing thresholds
> VlnPlot(object = ctrl, features.plot = c("nGene", "nUMI", "percent.mito"), nCol = 3)
>
> ############################ Set up stimulated object
>
> stim.data <- read.table(file = paste0(stim_samp.name, ".txt.gz"), header = TRUE, row.names
= 1, colClasses =c("character", rep("numeric", 1000)))
> #stim.data <- read.table(file = paste0(stim_samp.name, "_dge.txt.gz"), header = TRUE,
row.names = 1, colClasses =c("character", rep("numeric", 10000)))
```

```
> colnames(stim.data)<- paste0(colnames(stim.data),"_STIM")
>
>
> #Create Seurat object
> stim <- CreateSeuratObject(raw.data = stim.data, project = "IMMUNE_STIM", min.cells = 1,
min.genes = 500)
> stim@meta.data$stim <- "STIM"
>
> # mito genes to meta data
> mito.genes <- grep(pattern = "^mt-", x = rownames(x = stim@data), value = TRUE)
> percent.mito <- Matrix::colSums(stim@raw.data[mito.genes, ]) /
Matrix::colSums(stim@raw.data)
>
> # AddMetaData adds columns to object@meta.data, and is a great place to stash QC stats
> stim <- AddMetaData(object = stim, metadata = percent.mito, col.name = "percent.mito")
>
> #visualize for choosing thresholds
> VlnPlot(object = stim, features.plot = c("nGene", "nUMI", "percent.mito"), nCol = 3)
>
> # Chunk 4: thresh
> ctrl <- FilterCells(object = ctrl, subset.names = c("nGene", "nUMI", "percent.mito"),
low.thresholds = c(200,500, -Inf), high.thresholds = c(1400,2500, 0.02))
> ctrl <- NormalizeData(ctrl)
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> ctrl <- ScaleData(ctrl, display.progress = F)
>
> stim <- FilterCells(object = stim, subset.names = c("nGene", "nUMI", "percent.mito"),
low.thresholds = c(200,500, -Inf), high.thresholds = c(1400,2500, 0.017))
> stim <- NormalizeData(stim)
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> stim <- ScaleData(stim, display.progress = F)
>
>
> # Gene selection for input to CCA
> ctrl <- FindVariableGenes(ctrl, do.plot = F)
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> stim <- FindVariableGenes(stim, do.plot = F)
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
> g.1 <- head(rownames(ctrl@hvg.info), 1000)
> g.2 <- head(rownames(stim@hvg.info), 1000)
> genes.use <- unique(c(g.1, g.2))
> genes.use <- intersect(genes.use, rownames(ctrl@scale.data))
> genes.use <- intersect(genes.use, rownames(stim@scale.data))
>
> # Chunk 5: original identity
>
> write_ctrl=rep("ctrl", length(colnames(ctrl@data)))
> write.table(write_ctrl, file="ctrl_orig.ident.txt", sep="\t", eol ="\n",
row.names=colnames(ctrl@data))
>
```

```
> write_stim=rep("stim", length(colnames(stim@data)))
> write.table(write_stim, file="stim_orig.ident.txt", sep="\t", eol ="\n",
row.names=colnames(stim@data))
>
> # Chunk 6: singler ctrl
>
> ctrl_raw <- as.matrix(x=ctrl@data)
> singler_ctrl <- CreateSinglerSeuratObject(counts = ctrl_raw, annot="ctrl_orig.ident.txt",
project.name = ctrl_samp.name, min.genes = 500, min.cells = 1, technology = "Microwell-seq",
npca = 2, species = "Mouse", fine.tune = T)
[1] "CP1_dge_s"
[1] "Reading single-cell data..."
[1] "Create Seurat object..."
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Regressing out: nUMI
  |
==========================================================================================
==========================================| 100%
Time Elapsed:  11.408928155899 secs
Scaling data matrix
  |
==========================================================================================
==========================================| 100%
[1] "Creat SingleR object..."
[1] "Dimensions of counts data: 15232x708"
[1] "Annotating data with Immgen..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2751"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
==========================================================================================
============================| 100%, Elapsed 03:59
[1] "Number of DE genes:2751"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
==========================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Immgen (Main types)..."
[1] "Number of DE genes:1814"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
==========================================================================================
============================| 100%, Elapsed 00:36
[1] "Number of DE genes:1814"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
==========================================================================================
============================| 100%, Elapsed 00:02
[1] "Annotating data with Mouse-RNAseq..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2826"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
```

```
  |
================================================================================================
============================| 100%, Elapsed 00:35
[1] "Number of DE genes:2826"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Mouse-RNAseq (Main types)..."
[1] "Number of DE genes:2179"
[1] "Number of cells: 708"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:34
[1] "Number of DE genes:2179"
[1] "Number of clusters: 8"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
> singler_ctrl.new = convertSingleR2Browser(singler_ctrl)
> saveRDS(singler_ctrl.new, paste0(ctrl_samp.name, "_S4_ctrl.rds"))
> saveRDS(singler_ctrl, paste0(ctrl_samp.name, "_singleR_ctrl.rds"))
>
> # Chunk 7: singler stim
>
> stim_raw <- as.matrix(x=stim@data)
> singler_stim <- CreateSinglerSeuratObject(counts = stim_raw, annot= "stim_orig.ident.txt",
project.name = stim_samp.name, min.genes = 500, min.cells = 1, technology = "Microwell-seq",
species = "Mouse", npca = 10, fine.tune = T)
[1] "CS1_dge_s"
[1] "Reading single-cell data..."
[1] "Create Seurat object..."
Performing log-normalization
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene means
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Calculating gene variance to mean ratios
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Regressing out: nUMI
  |
================================================================================================
=========================================| 100%
Time Elapsed:  13.1719198226929 secs
Scaling data matrix
  |
================================================================================================
=========================================| 100%
[1] "Creat SingleR object..."
[1] "Dimensions of counts data: 15162x756"
[1] "Annotating data with Immgen..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2739"
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 03:17
[1] "Number of DE genes:2739"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
```

```
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Immgen (Main types)..."
[1] "Number of DE genes:1826"
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:37
[1] "Number of DE genes:1826"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:01
[1] "Annotating data with Mouse-RNAseq..."
[1] "Variable genes method: de"
[1] "Number of DE genes:2825"
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:34
[1] "Number of DE genes:2825"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:00
[1] "Annotating data with Mouse-RNAseq (Main types)..."
[1] "Number of DE genes:2186"
[1] "Number of cells: 756"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:32
[1] "Number of DE genes:2186"
[1] "Number of clusters: 4"
[1] "Fine-tuning round on top cell types (using 16 CPU cores):"
  |
================================================================================================
============================| 100%, Elapsed 00:00
> singler_stim.new = convertSingleR2Browser(singler_stim)
> saveRDS(singler_stim.new, paste0(stim_samp.name, "_S4_stim.rds"))
> saveRDS(singler_stim, paste0(stim_samp.name, "_singleR_stim.rds"))
>
>
> # Chunk 8: combine
>
> list_samples<- list(singler_ctrl, singler_stim)
> #list_xy <-
list(singler_ctrl[["seurat"]]@dr[["tsne"]]@cell.embeddings,singler_stim[["seurat"]]@dr[["tsn
e"]]@cell.embeddings)
>
> singler=SingleR.Combine(list_samples)
> saveRDS(singler, paste0(ctrl_samp.name,'_',stim_samp.name,"_tempfinal.rds"))
> singler_ctrl.stim.new = convertSingleR2Browser(singler, use.singler.cluster.annot = F)
Error in `row.names<-.data.frame`(`*tmp*`, value = value) :
  invalid 'row.names' length
> #saveRDS(singler_ctrl.stim.new, paste0(ctrl_samp.name,'_',stim_samp.name, '_final.rds'))
> traceback()
5: stop("invalid 'row.names' length")
4: `row.names<-.data.frame`(`*tmp*`, value = value)
3: `row.names<-`(`*tmp*`, value = value)
2: `rownames<-`(`*tmp*`, value = cell.names)
1: convertSingleR2Browser(singler, use.singler.cluster.annot = F)
```