Single Cell RNA Sequencing Worksheet2: Seurat

Anthor: Jesse Kurland, edited by Chris Ozeroff. July, 2023

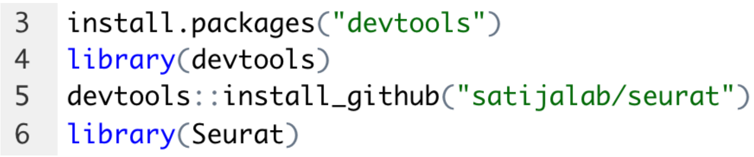
Reading in data and initial processing

1) Make a directory called cellranger_outputs_R_objects on your local machine

2) Rsync dpi7_filtered_feature_bc_matrix.zip, dpi4_filtered_feature_bc_matrix.zip, and final_myogenic_subset_seurat.RData from the AWS to this new directory. These can be found in /scratch/Shares/public/sread2023/data_files/day8/.

3) unzip the two cellranger output filtered matrix directories (dpi4 and dpi7).

4) Open Rstudio, and begin installing and loading necessary packages

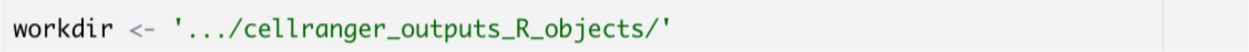


```r
install.packages("devtools")
library(devtools)
devtools::install_github("satijalab/seurat")
library(Seurat)
```

**If you get an error installing Seurat, use the following line of code instead of devtools::install_github("satijalab/seurat"):

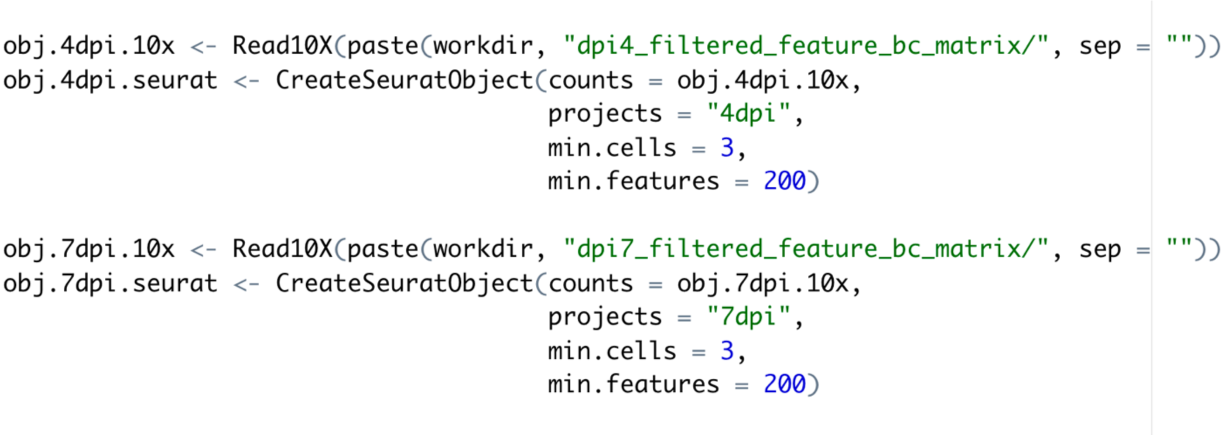
install.packages("https://cran.r-project.org/src/contrib/Archive/devtools/devtools_2.4.3.tar.gz", repos = NULL, type="source")

5) Set working directory with correct path.



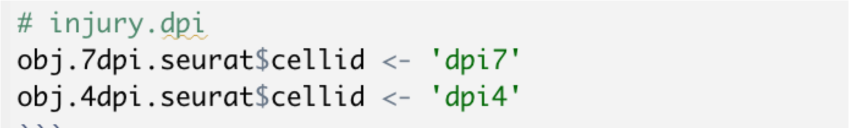
```r
workdir <- '.../cellranger_outputs_R_objects/'
```

6) Load in files from the "filtered_feature_bc_matrix" directories and use these to create a Seurat object



```r
obj.4dpi.10x <- Read10X(paste(workdir, "dpi4_filtered_feature_bc_matrix/", sep = ""))
obj.4dpi.seurat <- CreateSeuratObject(counts = obj.4dpi.10x,
                                      projects = "4dpi",
                                      min.cells = 3,
                                      min.features = 200)

obj.7dpi.10x <- Read10X(paste(workdir, "dpi7_filtered_feature_bc_matrix/", sep = ""))
obj.7dpi.seurat <- CreateSeuratObject(counts = obj.7dpi.10x,
                                      projects = "7dpi",
                                      min.cells = 3,
                                      min.features = 200)
```

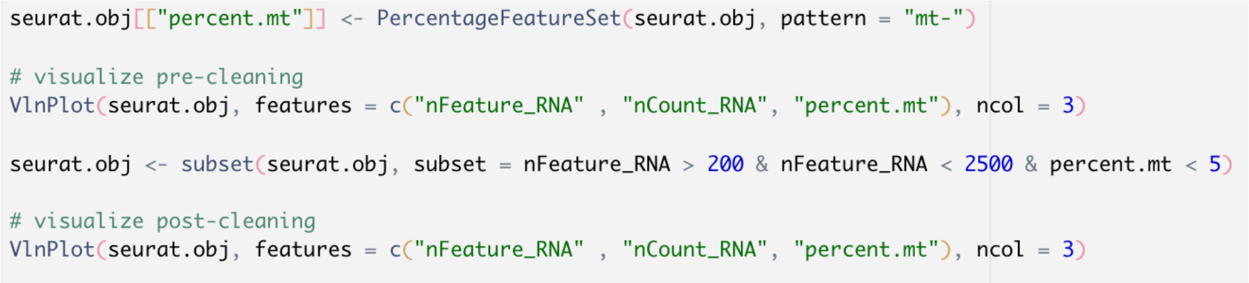
7) Add metadata columns to Seurat objects



```
# injury.dpi
obj.7dpi.seurat$cellid <- 'dpi7'
obj.4dpi.seurat$cellid <- 'dpi4'
```

8) Quality control

- Remove doublets (nFeature_RNA < 2500)
- Remove debris (nFeature_RNA > 200)
- Remove mitochondrial contamination (cells/nuclei with percent.mt < 5)

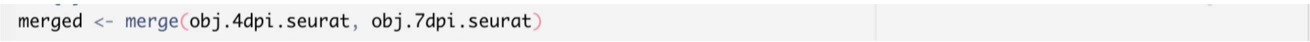


```
seurat.obj[["percent.mt"]] <- PercentageFeatureSet(seurat.obj, pattern = "mt-")

# visualize pre-cleaning
VlnPlot(seurat.obj, features = c("nFeature_RNA" , "nCount_RNA", "percent.mt"), ncol = 3)

seurat.obj <- subset(seurat.obj, subset = nFeature_RNA > 200 & nFeature_RNA < 2500 & percent.mt < 5)

# visualize post-cleaning
VlnPlot(seurat.obj, features = c("nFeature_RNA" , "nCount_RNA", "percent.mt"), ncol = 3)
```

**Note: this step must be done for each Seurat object, substituting obj.7dpi.seurat and obj.4dpi.seurat for searat.obj in each case.

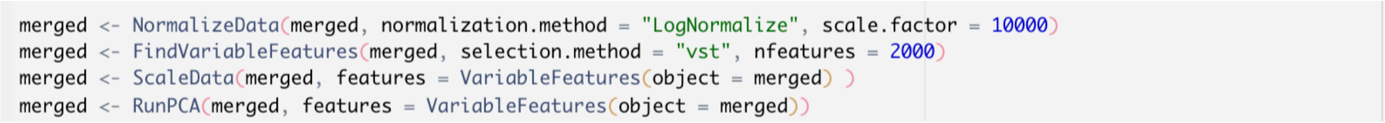
<u>Merging/Integration</u>

1) Merging Seurat objects (this has to be done prior to “integration” steps below)



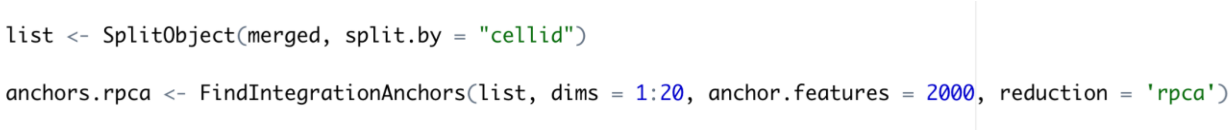
```
merged <- merge(obj.4dpi.seurat, obj.7dpi.seurat)
```

2) Processing merged object



```
merged <- NormalizeData(merged, normalization.method = "LogNormalize", scale.factor = 10000)
merged <- FindVariableFeatures(merged, selection.method = "vst", nfeatures = 2000)
merged <- ScaleData(merged, features = VariableFeatures(object = merged) )
merged <- RunPCA(merged, features = VariableFeatures(object = merged))
```

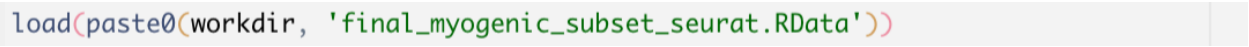
3) Integration using the rpca algorithim



```
list <- SplitObject(merged, split.by = "cellid")

anchors.rpca <- FindIntegrationAnchors(list, dims = 1:20, anchor.features = 2000, reduction = 'rpca')
```

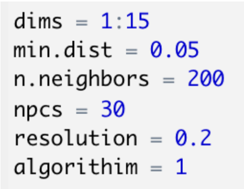
Clustering and visualization

1) Read in myogenic subset object



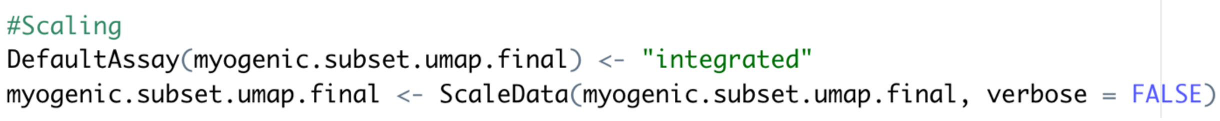
```
load(paste0(workdir, 'final_myogenic_subset_seurat.RData'))
```

2) Setting parameters for clustering and UMAP



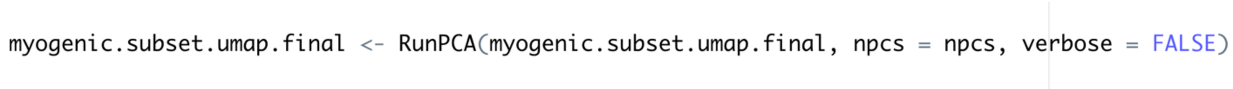
```
dims = 1:15
min.dist = 0.05
n.neighbors = 200
npcs = 30
resolution = 0.2
algorithim = 1
```

3) Scaling data



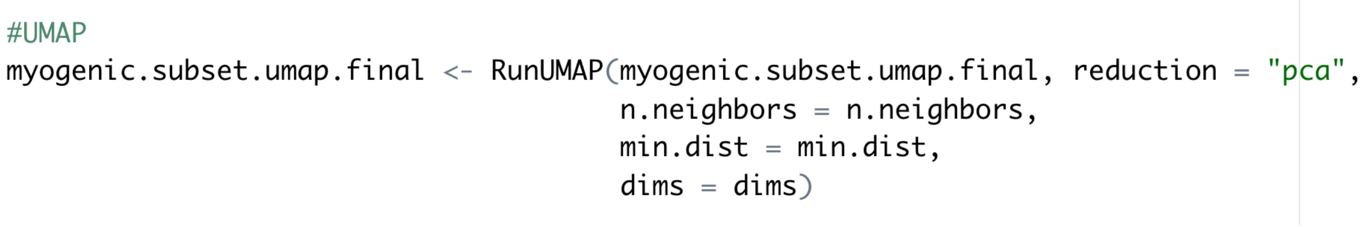
```
#Scaling
DefaultAssay(myogenic.subset.umap.final) <- "integrated"
myogenic.subset.umap.final <- ScaleData(myogenic.subset.umap.final, verbose = FALSE)
```

4) Conducting PCA reduction



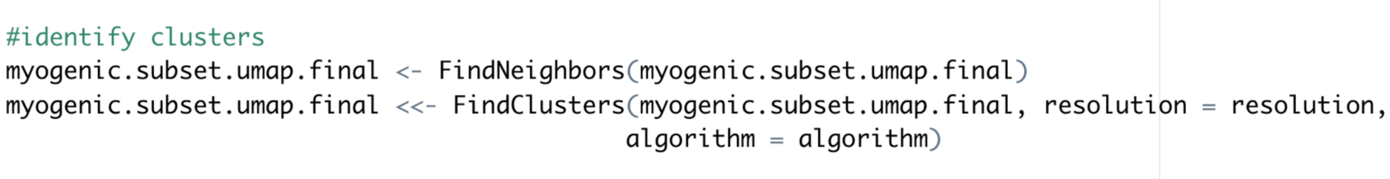
```
myogenic.subset.umap.final <- RunPCA(myogenic.subset.umap.final, npcs = npcs, verbose = FALSE)
```

5) Conducting UMAP reduction (see website for more information https://pair-code.github.io/understanding-umap/)



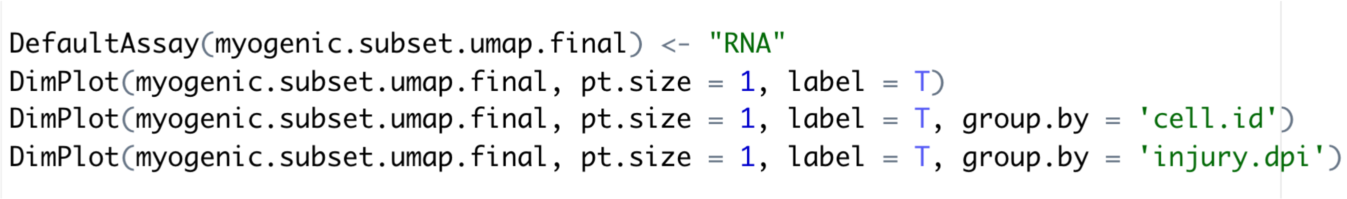
```
#UMAP
myogenic.subset.umap.final <- RunUMAP(myogenic.subset.umap.final, reduction = "pca",
                                       n.neighbors = n.neighbors,
                                       min.dist = min.dist,
                                       dims = dims)
```

6) Using the louvain algorithim to identify clusters from UMAP processed object



```
#identify clusters
myogenic.subset.umap.final <- FindNeighbors(myogenic.subset.umap.final)
myogenic.subset.umap.final <<- FindClusters(myogenic.subset.umap.final, resolution = resolution,
                                             algorithm = algorithm)
```

7) Plotting and visualizing clusters



```
DefaultAssay(myogenic.subset.umap.final) <- "RNA"
DimPlot(myogenic.subset.umap.final, pt.size = 1, label = T)
DimPlot(myogenic.subset.umap.final, pt.size = 1, label = T, group.by = 'cell.id')
DimPlot(myogenic.subset.umap.final, pt.size = 1, label = T, group.by = 'injury.dpi')
```