Supplementary Information

NS

2022-04-06

single-cell RNA sequecing analysis

Here, we use Seurat R package https://satijalab.org/seurat/ for: * performing quality control (QC) on scRNA-seq data from uninjured and stroke brain samples * visualizing the QC metrics * visualizing the expression of different cell-types specific marker genes * integrating scRNA-seq and spatial transcriptomics data to show the spatial patterns of select marker genes driven from scRNA-seq onto the Visium data of day10 post-stroke.

The results presented here are for the purpose of supplementary information. Please refer to Script 1-3 for more details on the exploration of single-cell RNA-seq data.

Load Libraries

```
library(Seurat)
library(tidyverse)
library(MAST)
library(Matrix)
library(knitr)
library(ggplot2)
library(cowplot)
library(patchwork)
library(viridisLite)
library(viridis)
library(tinytex)
```

Importing datasets

```
# Define the path to the main Directory
Uninj_data_dir <- "/Users/nickie/Desktop/Faiz_10Xdata/Uninjured"
Stroke_data_dir <- "/Users/nickie/Desktop/Faiz_10Xdata/Stroke"

# List the files whithin the Stroke and Uninjured subdirectory/ folders
## Should show barcodes.tsv.gz, features.tsv.gz, and matrix.mtx.gz
list.files(Uninj_data_dir, all.files = TRUE)
```

```
## [1] "."               ".."              "barcodes.tsv.gz" "features.tsv.gz"
## [5] "matrix.mtx.gz"
```

```
list.files(Stroke_data_dir, all.files = TRUE)
```

```
## [1] "."               ".."              "barcodes.tsv.gz" "features.tsv.gz"
## [5] "matrix.mtx.gz"
```

```
# Read in the count data
Uninj_data <- Read10X(data.dir = Uninj_data_dir)
Stroke_data <- Read10X(data.dir = Stroke_data_dir)
```

Create Seurat objects

```
# Use count data to make a Seurat object
Uninjured.seu.obj = CreateSeuratObject(counts = Uninj_data, min.cells = 3,
                                       min.features = 200, project = "Uninjured")
Stroke.seu.obj = CreateSeuratObject(counts = Stroke_data, min.cells = 3,
                                    min.features = 200, project = "Stroke")

# Check the Seurat objects
Uninjured.seu.obj
```

```
## An object of class Seurat
## 14784 features across 3092 samples within 1 assay
## Active assay: RNA (14784 features, 0 variable features)
```

```
Stroke.seu.obj
```

```
## An object of class Seurat
## 14899 features across 2603 samples within 1 assay
## Active assay: RNA (14899 features, 0 variable features)
```

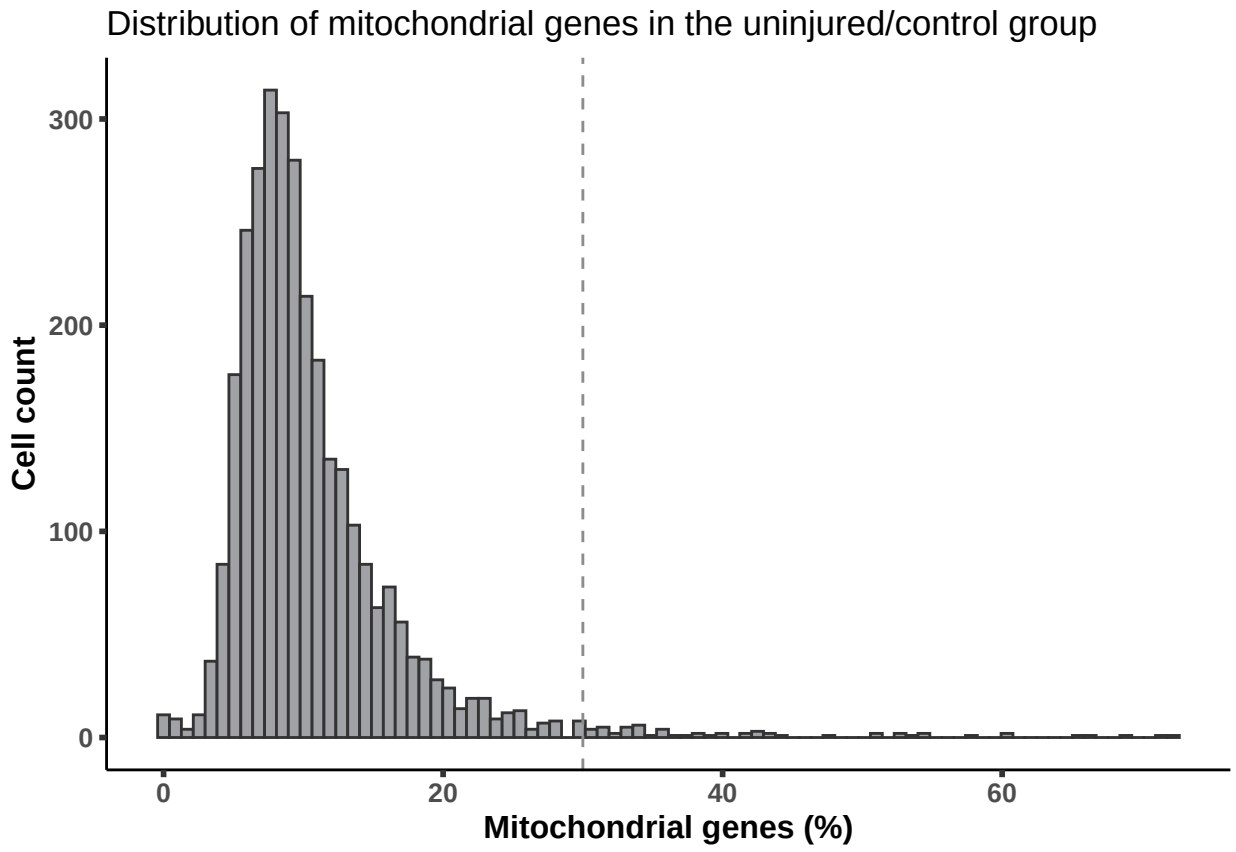
Pre-processing

```
# Calculate the percentage of reads that map to the mitochondrial genes
Uninjured.seu.obj[["percent.mt"]] <- PercentageFeatureSet(Uninjured.seu.obj,
                                                          pattern = "^mt-")

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

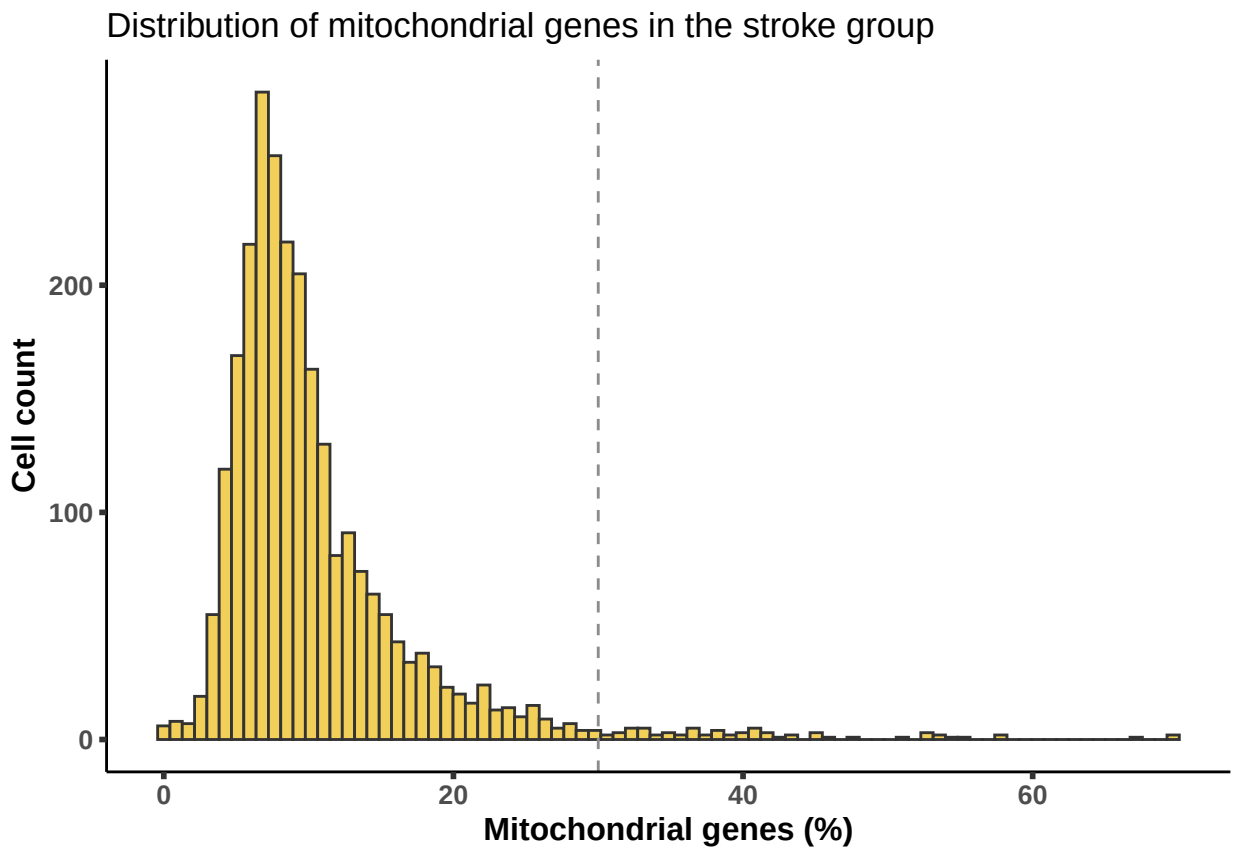
Visualize QC metrics, and use these to filter cells

Step 1) Get a sense of mitochondrial genes distribution

```
# Histogram plots
P1 <- Uninjured.seu.obj@meta.data %>%
  ggplot(aes(percent.mt)) +
  geom_histogram(binwidth = 0.85, fill="#a1a2a6", colour="grey20") +
  ggtitle("Distribution of mitochondrial genes in the uninjured/control group") +
  xlab("Mitochondrial genes (%)") + ylab("Cell count")+
  geom_vline(xintercept = 30, color="grey55", linetype= 2)+
  theme_classic()+
  theme(axis.title.x= element_text(face="bold", size=12),
        axis.title.y= element_text(face="bold", size=12),
        axis.ticks= element_line(size = 1),
        axis.text= element_text(face="bold", size=10),
        plot.title = element_text(colour="black"),
        panel.background = element_rect(fill = "NA"))

P2 <- Stroke.seu.obj@meta.data %>%
  ggplot(aes(percent.mt)) +
  geom_histogram(binwidth = 0.85, fill= "#f2cf59", colour="grey20") +
  ggtitle("Distribution of mitochondrial genes in the stroke group") +
  xlab("Mitochondrial genes (%)") + ylab("Cell count")+
  geom_vline(xintercept = 30, color="grey55", linetype= 2)+
  theme_classic()+
  theme(axis.title.x= element_text(face="bold", size=12),
        axis.title.y= element_text(face="bold", size=12),
        axis.ticks= element_line(size = 1),
        axis.text= element_text(face="bold", size=10),
        plot.title = element_text(colour="black"),
        panel.background = element_rect(fill = "NA"))

# Print the plots
P1
```


Distribution of mitochondrial genes in the uninjured/control group
Cell count
300
200
100
0
0
20
40
60
Mitochondrial genes (%)

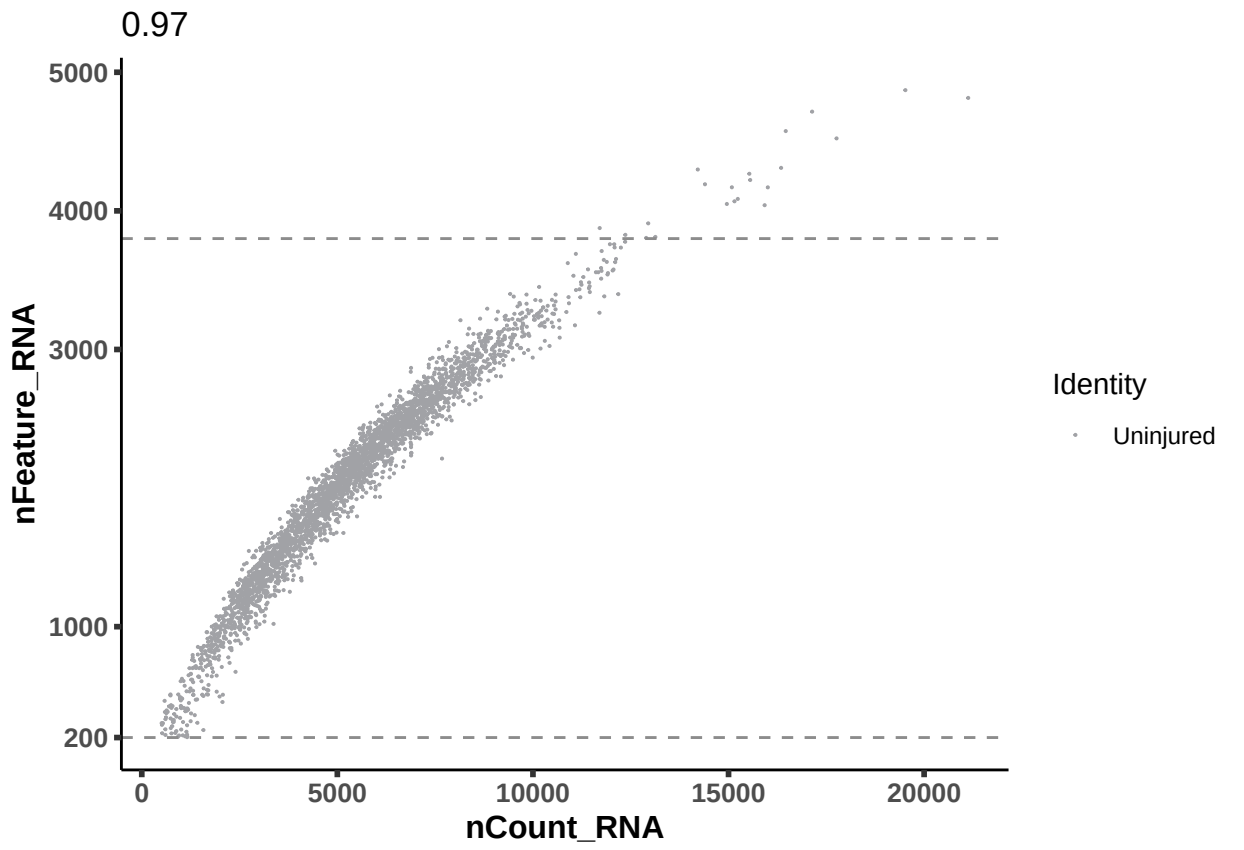
```
P2
```



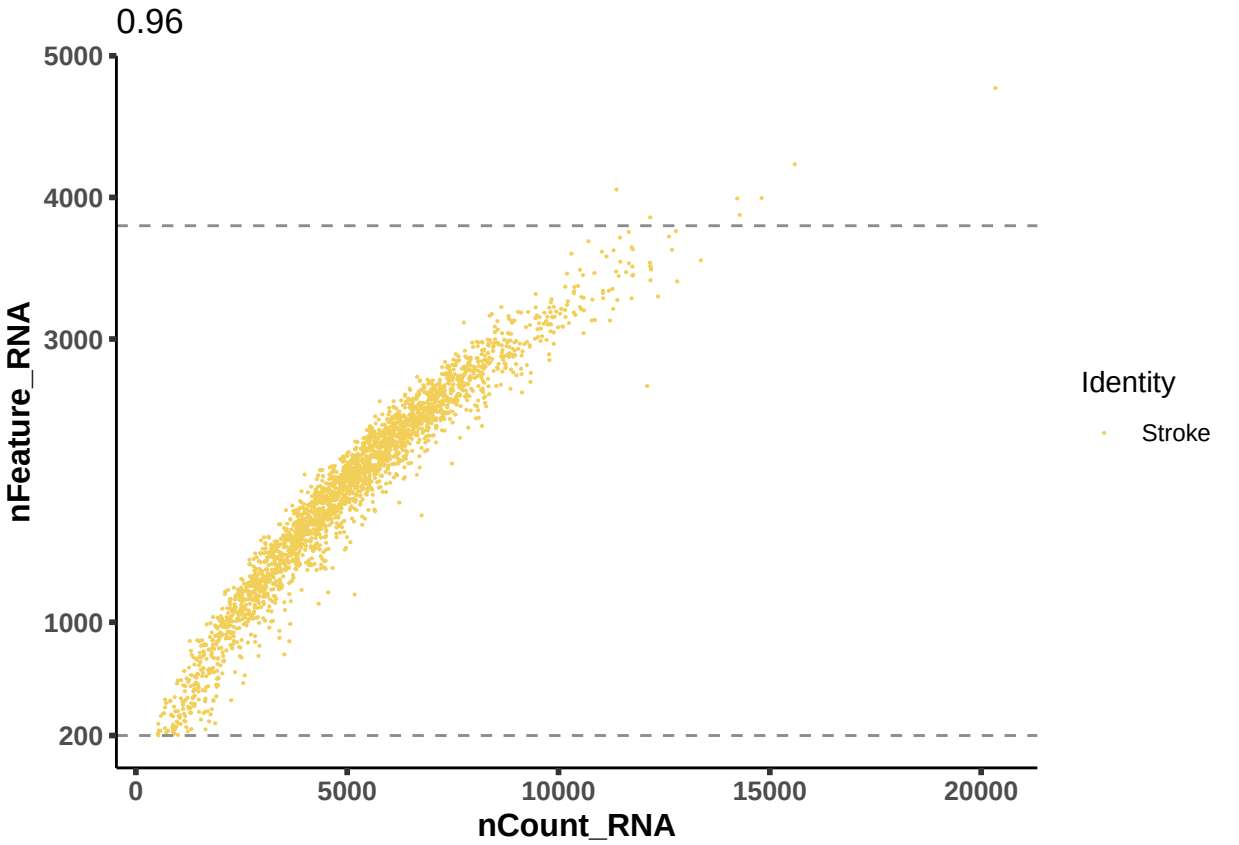
Step 2) Plot nCount_RNA and nFeature_RNA variables

```
P3 <- FeatureScatter(Uninjured.seu.obj, feature1 = "nCount_RNA",
                     feature2 = "nFeature_RNA",pt.size = 0.05,
                     cols = "#a1a2a6")+
  scale_y_continuous(breaks = c(200, 1000, 3000, 4000, 5000))+
  geom_hline(yintercept=c(200, 3800), linetype="dashed",
             color = "grey55", size=0.5)+
  theme_classic()+
  theme(axis.title.x= element_text(face="bold", size=12),
        axis.title.y= element_text(face="bold", size=12),
        axis.ticks= element_line(size = 1),
        axis.text= element_text(face="bold", size=10))



P4 <- FeatureScatter(Stroke.seu.obj, feature1 = "nCount_RNA",
                     feature2 = "nFeature_RNA", pt.size = 0.05,
                     cols = "#f2cf59" )+
  scale_y_continuous(breaks = c(200, 1000, 3000, 4000, 5000))+
  geom_hline(yintercept=c(200, 3800), linetype="dashed",
             color = "grey55", size=0.5)+
  theme_classic()+
  theme(axis.title.x= element_text(face="bold", size=12),
```

```
        axis.title.y= element_text(face="bold", size=12),
        axis.ticks= element_line(size = 1),
        axis.text= element_text(face="bold", size=10))



# Print the plots
P3
```




```
P4
```



Step 3) Filter cells that have >30% mitochondrial counts, and unique feature counts over 3800 or less than 200

```
Uninjured.seu.obj <- subset(Uninjured.seu.obj, subset = nFeature_RNA > 200 &
                              nFeature_RNA < 3800 & percent.mt < 30)

Stroke.seu.obj <- subset(Stroke.seu.obj, subset = nFeature_RNA > 200 &
                           nFeature_RNA < 3800 & percent.mt < 30)

# Check the results
Uninjured.seu.obj   #14784 features across 3009 samples within 1 assay
```

```
## An object of class Seurat
## 14784 features across 3009 samples within 1 assay
## Active assay: RNA (14784 features, 0 variable features)
```

```
Stroke.seu.obj #14899 features across 2529 samples within 1 assay
```

```
## An object of class Seurat
## 14899 features across 2529 samples within 1 assay
## Active assay: RNA (14899 features, 0 variable features)
```

Normalize each data

```
# Normalize counts data in seurat objects
Uninjured.seu.obj = NormalizeData(Uninjured.seu.obj,
                          normalization.method = "LogNormalize",
                          scale.factor = 1000000)

Stroke.seu.obj = NormalizeData(Stroke.seu.obj,
                                  normalization.method = "LogNormalize",
                                  scale.factor = 1000000)
```

Merge Uninjured and Stroke datasets for unbiased downstream analysis

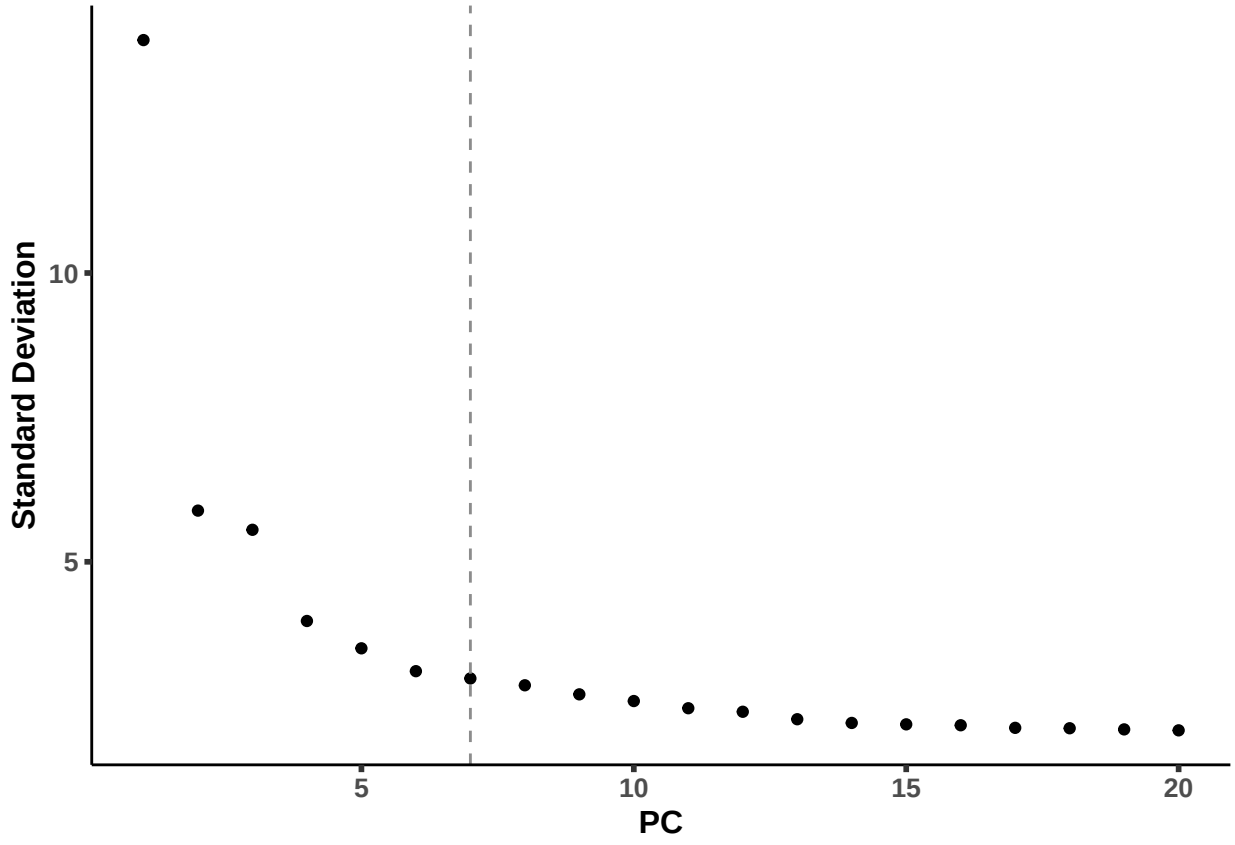
```
# Merge the two Seurat.objects
seu.combined <- merge(x=Uninjured.seu.obj, y = Stroke.seu.obj,
                      add.cell.ids = c("UN", "S"),
                      project = "UNandS")

seu.combined
```

```
## An object of class Seurat
## 15472 features across 5538 samples within 1 assay
## Active assay: RNA (15472 features, 0 variable features)
```

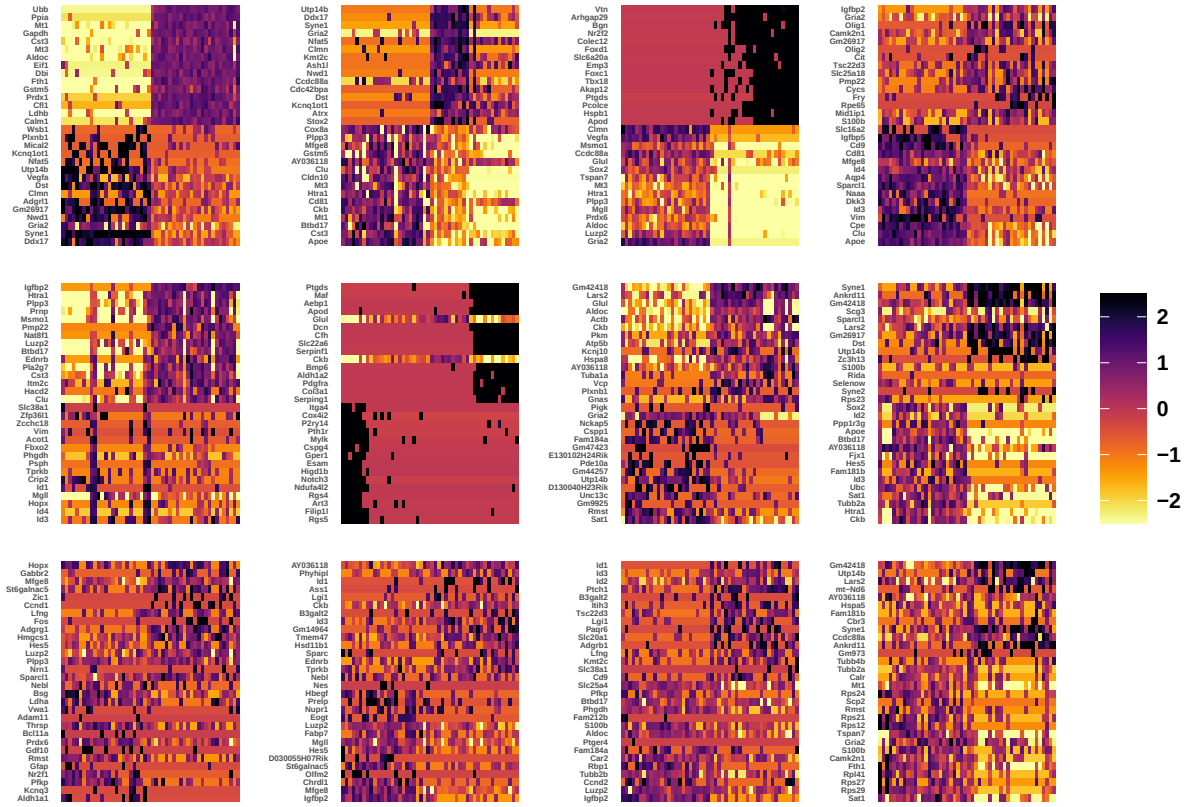
Standard workflow

```
#1) Normalization
seu.combined <- NormalizeData(seu.combined, normalization.method = "LogNormalize",
                              scale.factor = 10000)

#2) Centering and Scaling data (for all genes)
set.seed(42)
all.genes_UNandS <- rownames(seu.combined)
seu.combined <- ScaleData(seu.combined, features = all.genes_UNandS)

#3) Find variable features
seu.combined <- FindVariableFeatures(seu.combined,
                                     selection.method = "vst", nfeatures = 7800) #50% VF

#4) Principle component analysis
seu.combined <- RunPCA(seu.combined, npcs = 30,
                       features = VariableFeatures(object = seu.combined))

# Check PCs

P5 <- ElbowPlot(object = seu.combined, reduction = "pca")+
  geom_vline(xintercept=7, linetype="dashed", color = "grey55", size=0.5)+
  theme_classic()+
  theme(axis.title.x= element_text(face="bold", size=12),
```

```
        axis.title.y= element_text(face="bold", size=12),
        axis.ticks= element_line(size = 1),
        axis.text= element_text(face="bold", size=10))

P6 <- DimHeatmap(seu.combined, dims = 1:12, cells = 50,
                 balanced = TRUE, ncol = 4,fast = FALSE) &
  scale_fill_viridis(option="inferno", direction=-1)&
  theme(text = element_text(size = 3),
        axis.title.x= element_text(face="bold", size=3),
        axis.title.y= element_text(face="bold", size=3),
        axis.text= element_text(face="bold", size=3),
        legend.text= element_text(face="bold", size=8),
        legend.title= element_text(face="bold", size=8))


# Print the plots
P5 # The elbow occurs at the 7th PCs
```




```
P6
```



Dimential reduction

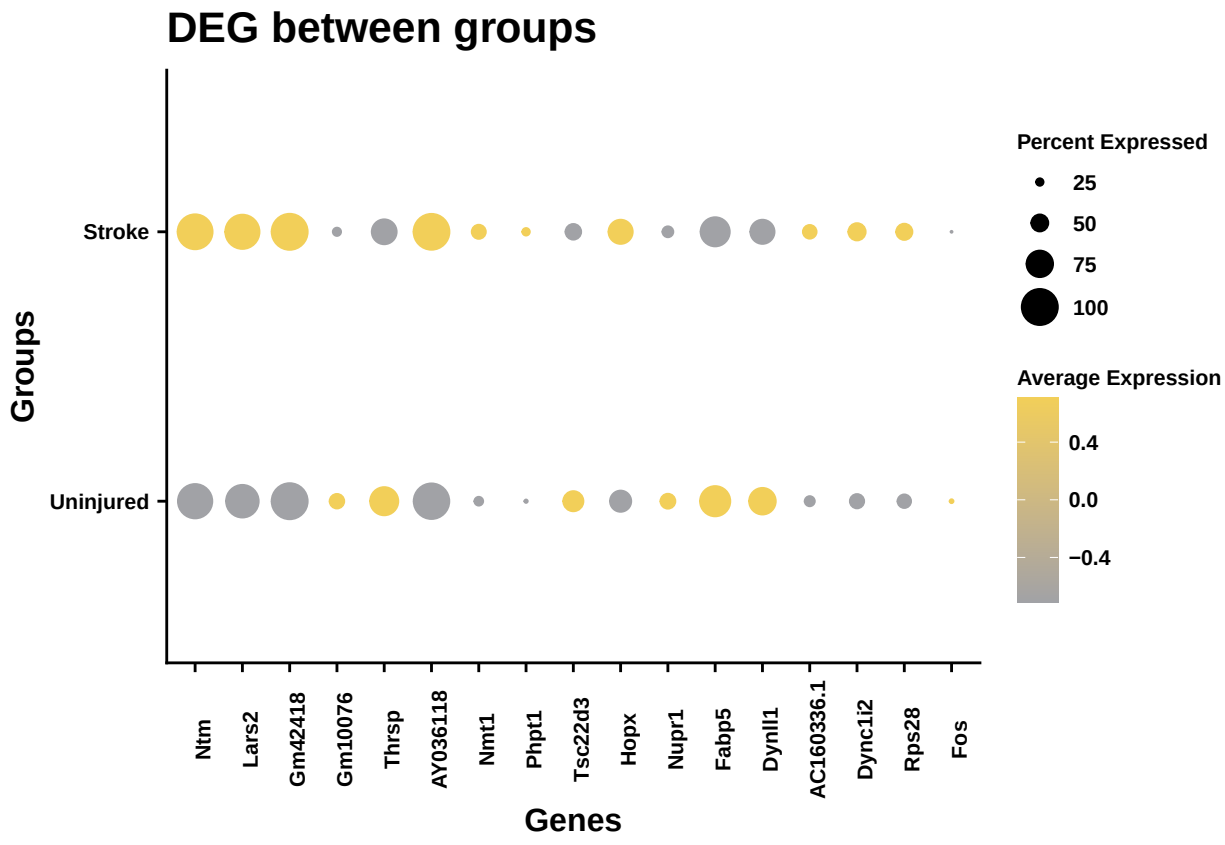
```
# Find neighbours
seu.combined <- FindNeighbors(seu.combined, dims = 1:10)
seu.combined <- FindClusters(seu.combined, resolution = 0.5)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 5538
## Number of edges: 175394
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8086
## Number of communities: 8
## Elapsed time: 0 seconds
```

```
# Non-Linear dimential reduction
seu.combined <- RunUMAP(seu.combined, reduction = "pca", dims = 1:10)
seu.combined = RunTSNE(seu.combined, reduction = "pca", dims = 1:10)
```

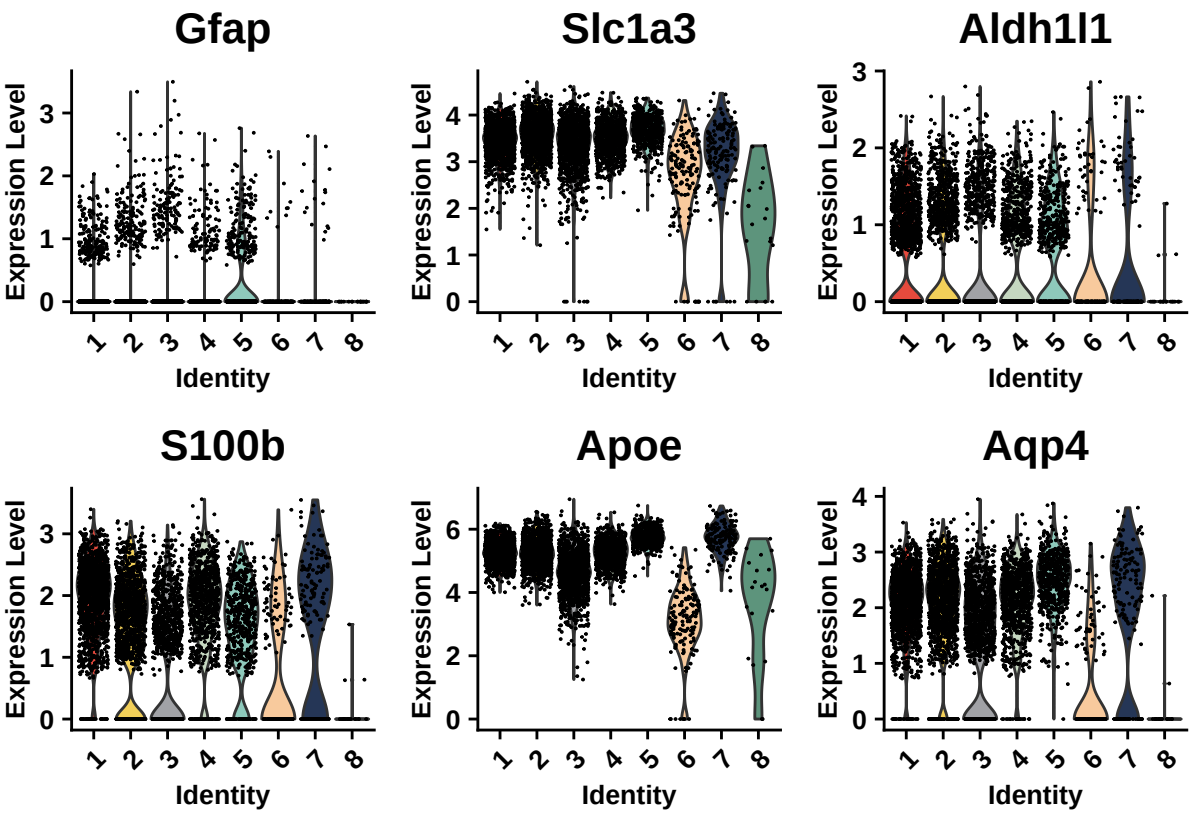
Identify differentially expressed genes across conditions

```
Idents(seu.combined) <- "orig.ident"

UNvsS <- FindMarkers(seu.combined, group.by = 'orig.ident',
                     ident.1 = "Uninjured", ident.2 = "Stroke") %>% rownames_to_column(var="gene")

# Visualize
P7 <- DotPlot(seu.combined, features = UNvsS$gene, assay="RNA",
              cols= c("#a1a2a6", "#f2cf59"))+
  scale_fill_viridis(option="inferno", direction=-1, guide = "colourbar")+
  labs(title = "DEG between groups",x = "Genes", y = "Groups")+
  theme(axis.title.x= element_text(face="bold", size=12),
        axis.title.y= element_text(face="bold", size=12),
        axis.text= element_text(face="bold", size=8),
        axis.text.x = element_text(angle=90),
        legend.text= element_text(face="bold", size=8),
        legend.title= element_text(face="bold", size=8),
        legend.box= "vertical")

P7
```



Visualize cell type specific markers in clusters

```
# Define markers
Astrocyte.general.markers <- c('Gfap', 'Slc1a3', 'Aldh1l1', 'S100b', 'Apoe', 'Aqp4')
Protoplasmic.astrocytes.markers <- c('Mfge8', 'Slc1a3', 'Slc1a2', 'Nupr1', 'Thrsp')
Fibrous.astrocytes.markers <- c('Gfap', 'Vim', 'Id3', 'Cd9', 'Fos')
Upper.cortical.layer.astrocytes.markers <- c('Mfge8', 'Igfbp2')

# Markers for driving a fate in astrocytes:
Fd.markers <- c('Apoe', 'Ccnd1', 'Lgal3bp', 'Serpina3n','C4b')

# Non reactive astrocytes markers:
NR.astrocytes.markers <- c('S100a10', 'Aldoc')

# Astrocyte markers- in response to microglia/interferons (Mi):
Mi.markers <- c('Igtb','Lgal3bp', 'Tnfrsf1a','Stat1')

# Astrocyte - types inducing oligodendrocyte migration (Om):
Om.markers <- c('Bmp4', 'Timp1', 'Gab1', 'Pdgfp')

# ECM remodeling markers:
ECM.markers <- c('S100a4', 'Tgfb1', 'Timp1')

# Astrocytes markers involved in angiogenesis and blood pressure:
Ang.markers <- c('Fbln5', 'Agt')

# Endothelial cells markers:
En.markers <- c('Cd44', 'Cd34',  'Entpd1','Icam1', 'Itgb1','Vcam1')


# Plot markers expression levels in clusters
## Make clusters start at 1
seu.combined$new_clusters <-
  as.factor(as.numeric(as.character(seu.combined$seurat_clusters))+1)

## Set idents
Idents(seu.combined) = "new_clusters"

## Set colors
my.cols = c ('#e64a3d',  '#f2cf59', '#a1a2a6', '#c5d7c0',
             '#8ec9bb', '#f8ca9d', '#253656', '#5d947c', "#000000")

P8 <- VlnPlot(seu.combined, features = Astrocyte.general.markers,
              ncol= 3, pt.size =0.01) &
  scale_fill_manual(values = my.cols) &
  theme(axis.title.x= element_text(face="bold", size=10),
        axis.title.y= element_text(face="bold", size=10),
        axis.text= element_text(face="bold", size=10),
        axis.text.x = element_text(angle=45))

P8 # you may use other cell-type markers defined above
```



Visualize selected DEG between Penumbra and non-penumbra astrocytes clusters on Visium data

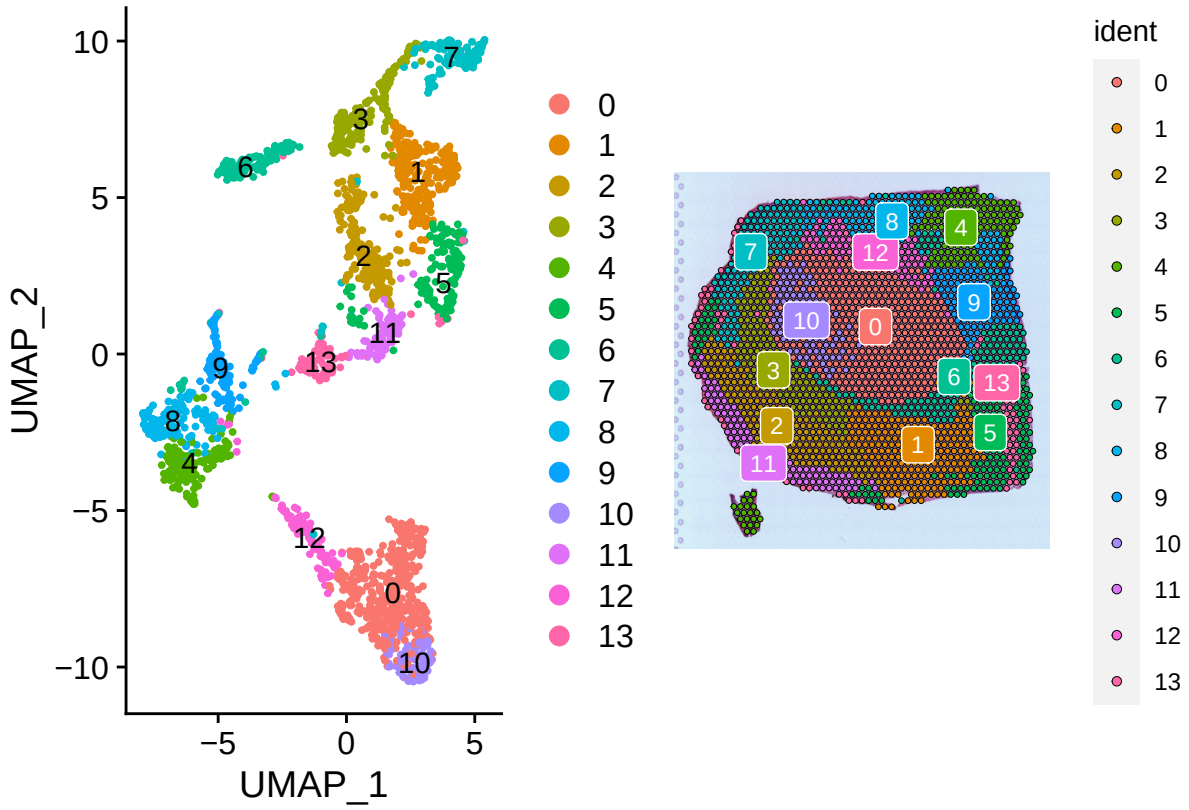
Step 1) Import and analyze the day10_Visium_data

```
# Define the path to the visium data directory
data_dir_d10 <- "/Users/nickie/Desktop/Faiz_Visiumdata/Faiz_Maryam__V10A06-087-D1/"

# List the files
list.files(data_dir_d10, all.files = TRUE)
```

```
## [1] "."                              ".."
## [3] ".DS_Store"                      "filtered_feature_bc_matrix.h5"
## [5] "scalefactors_json.json"         "spatial"
## [7] "tissue_hires_image.png"         "tissue_positions_list.csv"
```

```
# Read the spatial data using Seurat
FV.d10 <- Load10X_Spatial(data.dir = data_dir_d10)

# Normalize the data
##(sctransform normalizes the data, detects high-variance features,
## and stores the data in the SCT assay)
FV.d10 <- SCTransform(FV.d10, assay = "Spatial", verbose = FALSE)
```

```
# Dimensionality reduction, clustering, and visualization
FV.d10<- RunPCA(FV.d10, assay = "SCT", verbose = FALSE)
FV.d10<- FindNeighbors(FV.d10, reduction = "pca", dims = 1:30)
FV.d10<- FindClusters(FV.d10, verbose = FALSE)
FV.d10<- RunUMAP(FV.d10, reduction = "pca", dims = 1:30)

# Visualize clusters
P9 <- DimPlot(FV.d10, reduction = "umap", label = TRUE)
P10 <- SpatialDimPlot(FV.d10, label = TRUE, label.size = 3)
P9 + P10
```



Step 2) Define genes to be mapped

```
# Group 1) Genes upregulated in cluster 7 astrocytes with
# expression patterns localized around the injury site on Visium data
Group1.genes <- c('Apoe', 'Cd81', 'Lamp1', 'Ftl1' ,'Tpt1',
                  'Ctsl', 'Fau', 'Dbi', 'Eef1a1', 'Fabp5')

# Group 2) Genes upregulated in cluster 7 astrocytes; though,
# with inverse expression patterns on Visium data
# (i.e., more localized outside the injury site)
Group2.genes <- c('Mt1', 'Aldoc', 'Cox8a', 'Clu',
                  'Ppia', 'Gstm1', 'Cldn10', 'Mfge8')
```

Step3) Visualize selected genes on both 10X single-cell and visium datasets

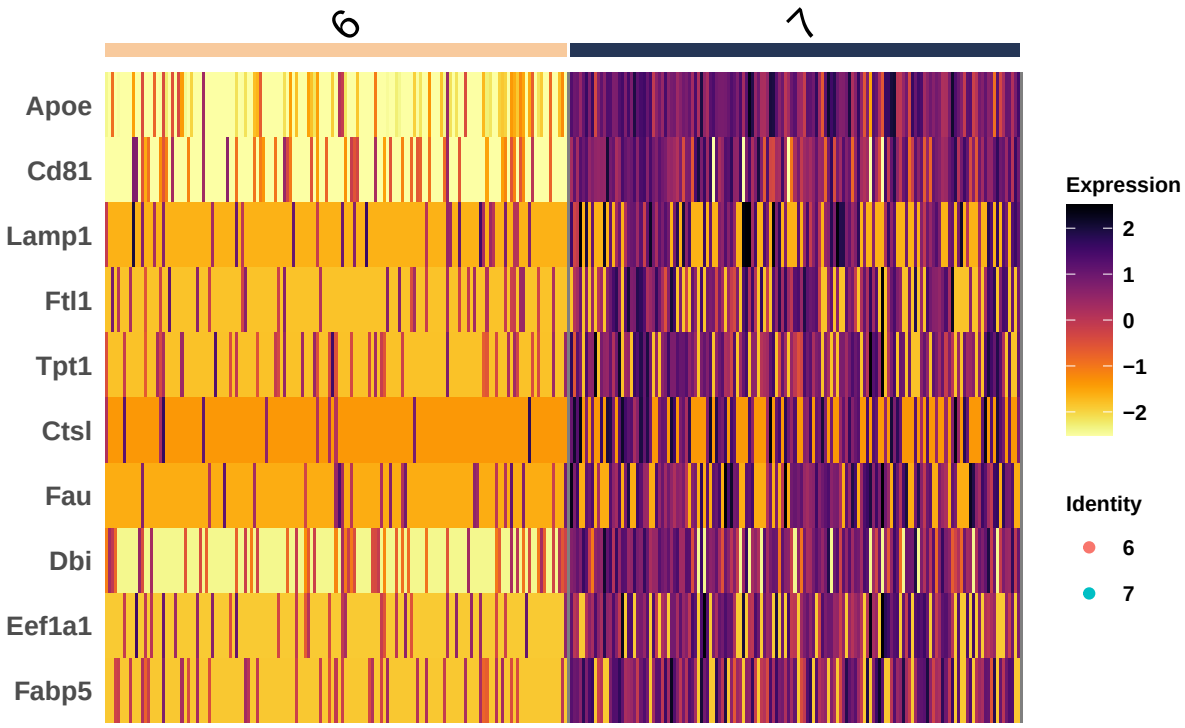
```
#1) Subset Seurat object for the two clusters of interest
seu.cluster6and7 <- subset(seu.combined, subset=new_clusters %in%c("6","7"))

#2) Heatmap expression levels in the single-cell data
P11 <- DoHeatmap(object = seu.cluster6and7, features = Group1.genes,
                 label = TRUE, group.colors= c('#f8ca9d','#253656')) +
  scale_fill_viridis(option="inferno", direction=-1) +
  theme(text = element_text(size = 3),
        axis.title.x= element_text(face="bold", size=10),
        axis.title.y= element_text(face="bold", size=10),
        axis.text= element_text(face="bold", size=10),
        legend.text= element_text(face="bold", size=8),
        legend.title= element_text(face="bold", size=8))
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

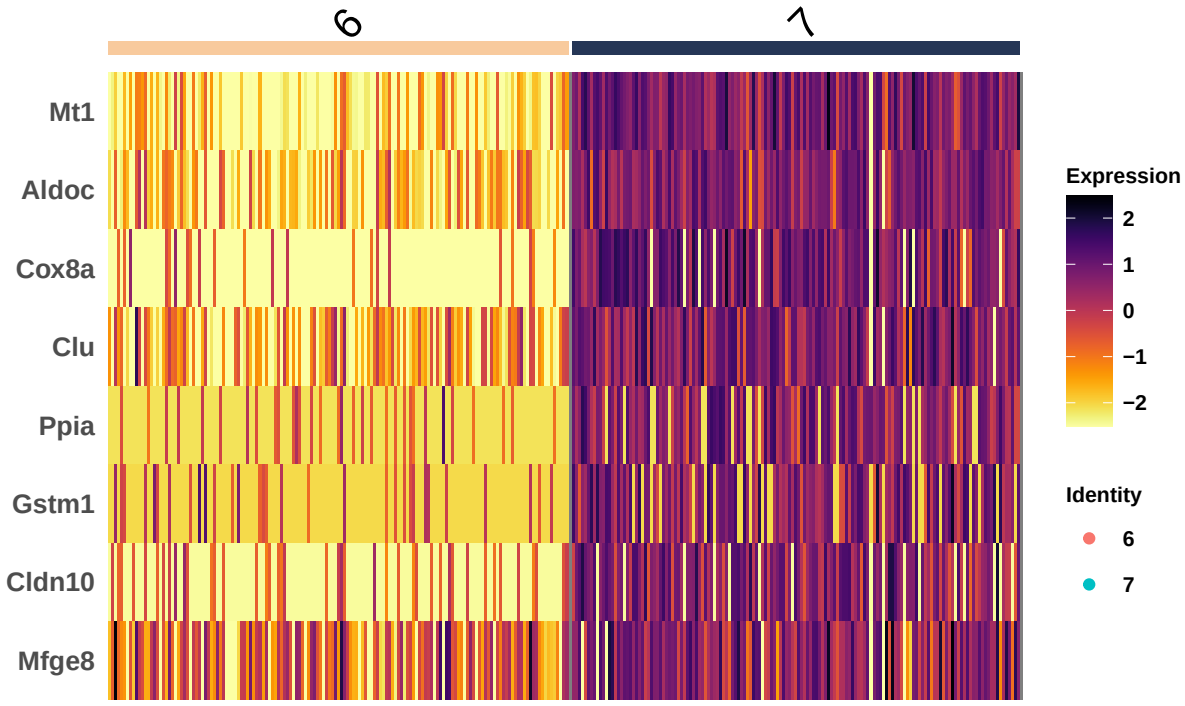
```
P12 <- DoHeatmap(object = seu.cluster6and7, features = Group2.genes,
                 label = TRUE, group.colors= c('#f8ca9d','#253656')) +
  scale_fill_viridis(option="inferno", direction=-1) +
  theme(text = element_text(size = 3),
        axis.title.x= element_text(face="bold", size=10),
        axis.title.y= element_text(face="bold", size=10),
        axis.text= element_text(face="bold", size=10),
        legend.text= element_text(face="bold", size=8),
        legend.title= element_text(face="bold", size=8))
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

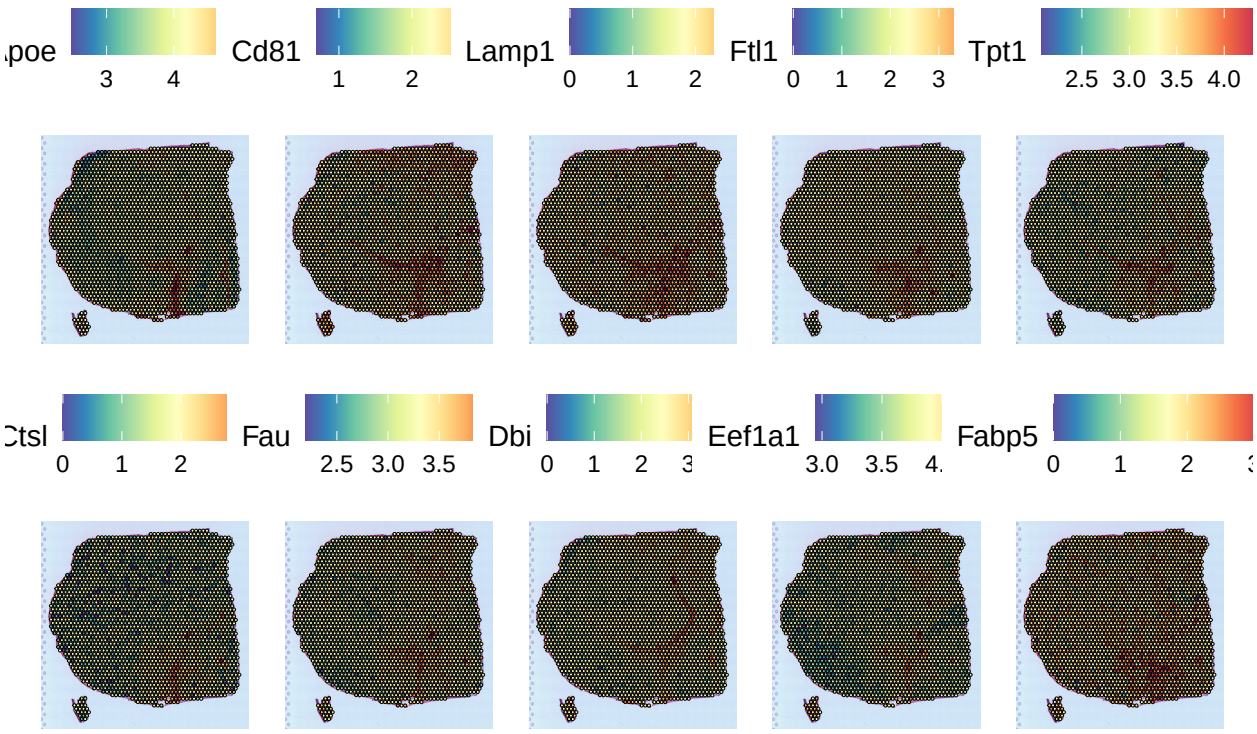
```
#3) Spatial mapping on Visium data (day10)
P13 <- SpatialFeaturePlot(FV.d10, features = Group1.genes, ncol = 5)
P14 <- SpatialFeaturePlot(FV.d10, features = Group2.genes, ncol = 4)

# Print
P11
```


6
7
Apoe
Cd81
Lamp1
Ftl1
Tpt1
Ctsl
Fau
Dbi
Eef1a1
Fabp5
Expression
2
1
0
−1
−2
Identity
6
7

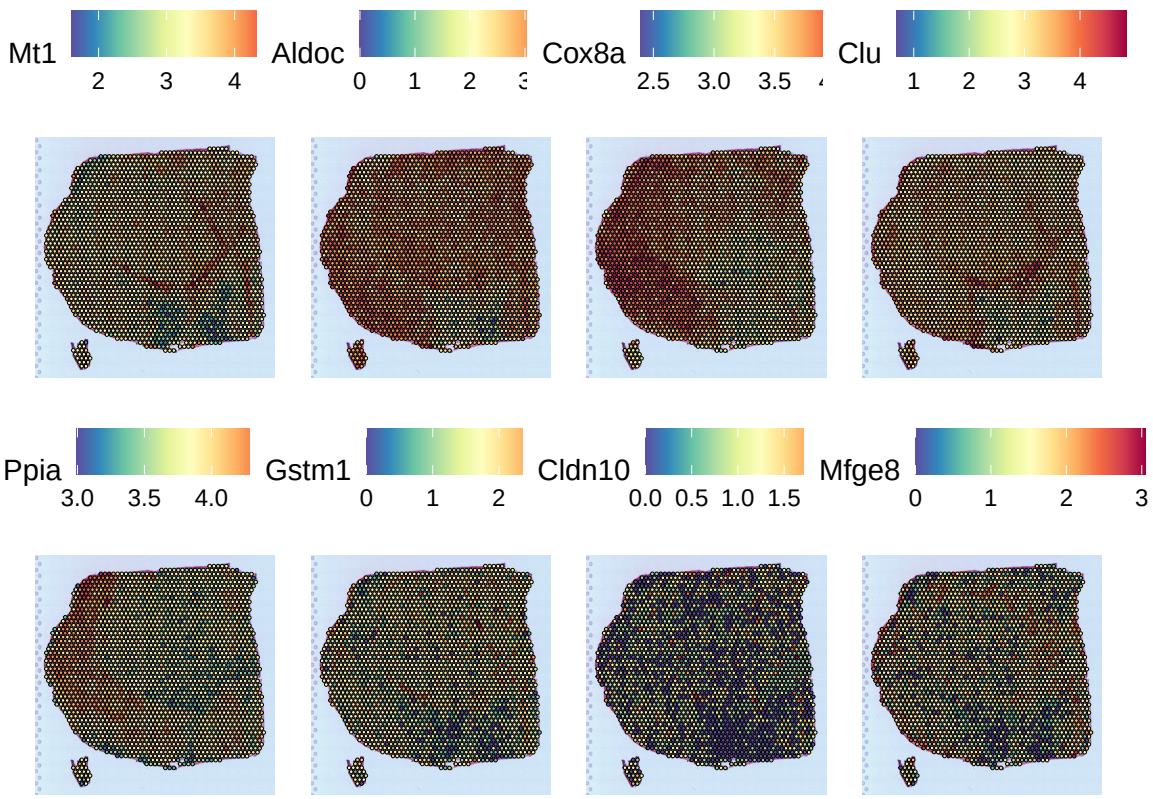
```
P12
```


6
7
Mt1
Aldoc
Cox8a
Clu
Ppia
Gstm1
Cldn10
Mfge8
Expression
2
1
0
−1
−2
Identity
6
7

```
P13
```


poe
3 4
Cd81
1 2
Lamp1
0 1 2
Ftl1
0 1 2 3
Tpt1
2.5 3.0 3.5 4.0
Ctsl
0 1 2
Fau
2.5 3.0 3.5
Dbi
0 1 2 3
Eef1a1
3.0 3.5 4.
Fabp5
0 1 2 3

```
P14
```



Convert this file into a jupyter file

```
# Install the package
devtools::install_github("mkearney/rmd2jupyter")
library(rmd2jupyter)

# Convert
rmd2jupyter("Script4_Codes.for.Supplementary.Information.Rmd")
## you should see this file in your working directory
```