Disease associated microglia and disease associated astrocyte markers expression changes after immune challenge in human ApoE mouse models

Hector Zepeda

Final Project

ABSTRACT

This report outlines the analysis of single-cell RNA sequencing data from microglial cells and astrocytes in response to different conditions, specifically APOE genotype variations and treatment with lipopolysaccharide (LPS) or saline. The analysis aims to understand how APOE genotype and inflammatory challenges influence microglial immunometabolism. The Seurat package in R was utilized for quality control, normalization, feature selection, dimensionality reduction, clustering, and differential gene expression analysis. The results reveal distinct cellular clusters and differentially expressed genes associated with APOE genotype and treatment conditions. This report provides insights into the molecular mechanisms underlying microglial and astrocytic responses to APOE genotype and inflammatory stimuli. Thus, the analyzed dataset counts with four total groups to analyze: ApoE3+Saline Injection, ApoE3+LPS, ApoE4+Saline Injection and ApoE4+LPS.

INTRODUCTION

Microglia and astrocytes are crucial players in maintaining CNS homeostasis and responding to various pathological stimuli, including inflammation and neurodegeneration. Microglia, the resident immune cells of the CNS, are involved in immune surveillance, synaptic pruning, and neuroinflammatory responses. Astrocytes, on the other hand, provide metabolic and structural support to neurons and actively participate in modulating synaptic function and inflammation. Dysregulation of microglia and astrocyte functions has been implicated in numerous neurological disorders, including Alzheimer's disease (AD), Parkinson's disease (PD), and multiple sclerosis (MS). Apolipoprotein E (ApoE), a lipid-binding protein predominantly

produced by astrocytes in the CNS, plays a critical role in lipid metabolism, synaptic plasticity, and immune regulation. There are three common isoforms of ApoE: ApoE2, ApoE3, and ApoE4. Among these isoforms, ApoE4 has been strongly associated with an increased risk of developing AD, while ApoE3 has been suggested to maintain a homeostatic environment. However, the molecular mechanisms underlying these isoform-specific effects remain elusive. In this study, we leverage single-cell RNA sequencing (scRNA-seq) data to investigate how microglia and astrocytes respond to different ApoE isoforms and environmental challenges such as LPS-induced inflammation and saline treatment. By analyzing the transcriptional profiles of these cell types under various conditions, we aim to unravel the intricate molecular mechanisms governing their immunometabolic responses and their potential implications for neurodegenerative diseases in a neuroinflammatory context.

METHODOLOGY

Seurat

To conduct a comprehensive analysis of the single-cell RNA sequencing (scRNA-seq) data for this project, I employed Seurat, a robust computational framework tailored for scRNA-seq analysis. Seurat streamlines the analysis workflow by integrating various algorithms and statistical methods into a cohesive toolkit, facilitating the exploration and interpretation of complex gene expression datasets. The first step in analysis pipeline using Seurat, involved importing the raw gene expression data into R, the programming environment used for implementing Seurat. This process includes reading in the gene count matrix, which contains information on the expression levels of individual genes across the sampled cells.

Data Processing

Before any analysis, it's crucial to normalize the gene expression data to ensure that differences in sequencing depth or cell size don't confound the analysis. Normalization adjusts the expression values for each gene across cells to have similar distributions, typically by scaling or transforming the data. However, not all genes are equally informative for characterizing cell types or states. By identifying highly variable features, we focus on the genes that exhibit the most significant changes across cells, enabling us to capture the most relevant biological variation.

Dimensionality Reduction

Principal component analysis (PCA) is a statistical technique used to reduce the dimensionality of high-dimensional data while preserving most of the variance. It achieves this by transforming the original variables (genes in this case) into a new set of variables, called principal components, which are linear combinations of the original variables. These principal components are ordered by the amount of variance they explain, allowing us to visualize and analyze the data

in a lower-dimensional space. Once the dimensionality is reduced, clustering algorithms are applied to group cells with similar gene expression profiles into clusters. Clustering helps identify distinct cell populations or states within the dataset, allowing us to characterize cell heterogeneity and infer cell types or subtypes.

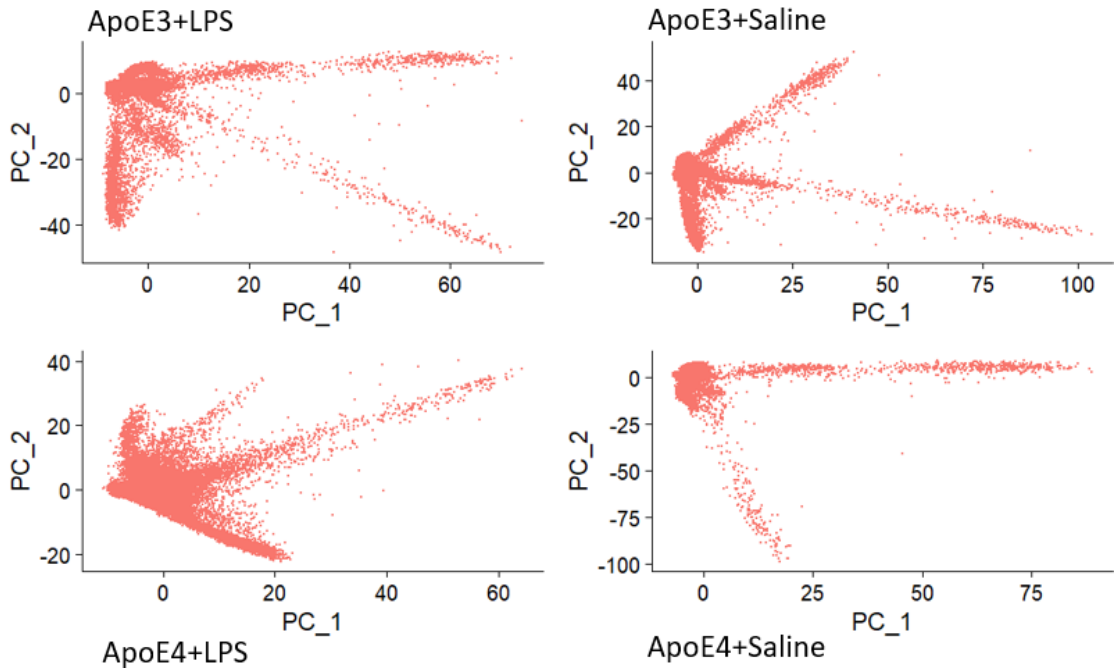

Figure 1: Principal Component Analysis for Four Groups.

So then once the principal components were acquired, a Uniform Manifold Approximation and Projection (UMAP) was done across the cells based on their features and proximity to others. A UMAP is a nonlinear dimensionality reduction technique that preserves local and global structure in high-dimensional data. Unlike PCA, which focuses on linear relationships, UMAP captures complex, nonlinear relationships between cells in the original high-dimensional space and projects them onto a lower-dimensional space. This technique is particularly useful for visualizing high-dimensional single-cell data in two dimensions while maintaining the overall structure of the data.

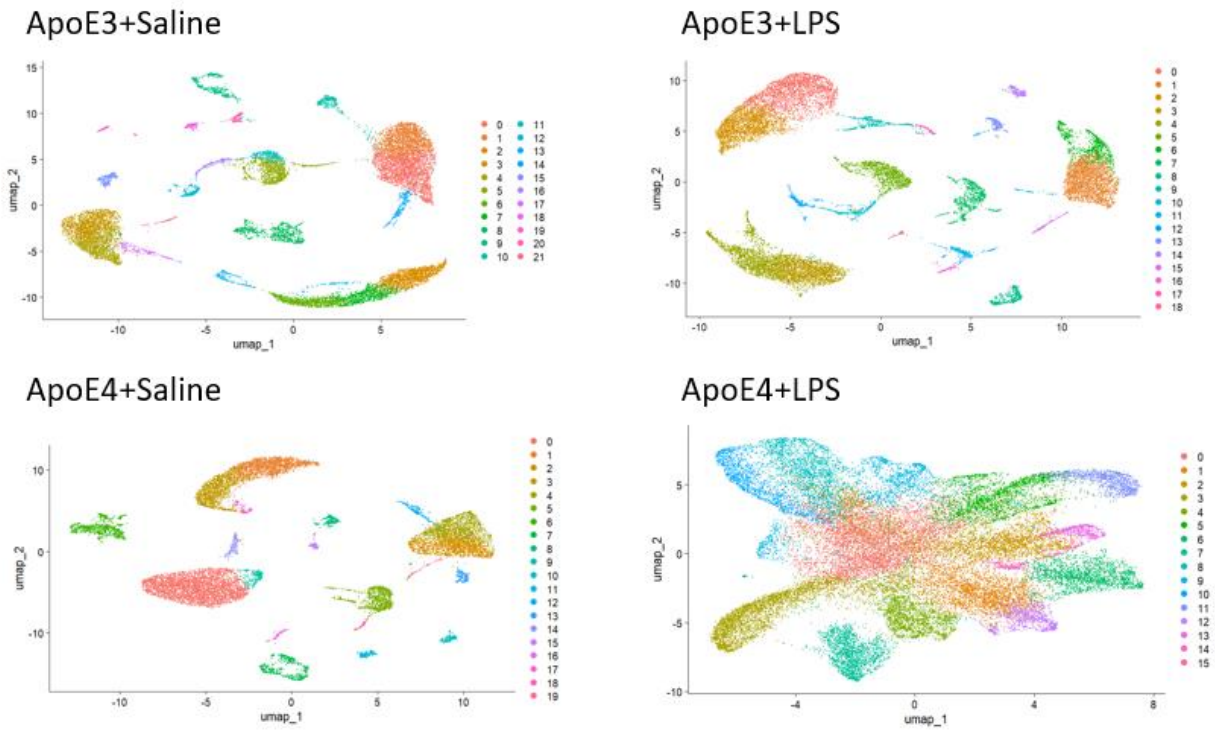

Figure 2: UMAP and Clustering for Four Groups.

After preprocessing the data, including normalization and identification of highly variable features, the next step involved merging the four Seurat objects corresponding to the distinct experimental conditions or groups of study into a single Seurat object. This consolidation was supposed to facilitate comprehensive analysis and comparison across all groups. Subsequently, principal component analysis (PCA) was performed to reduce the dimensionality of the high-dimensional gene expression data, preserving the majority of the variance while simplifying visualization and interpretation. Following PCA, a t-distributed stochastic neighbor embedding (tSNE) analysis was conducted to further visualize the data in a lower-dimensional space. The decision to utilize tSNE for dimensionality reduction was due to the prior study referenced in the research paper from which this data was sourced. In their analysis of the four groups together, the researchers found that tSNE provided superior clustering results compared to other dimensionality reduction techniques, making it the preferred method for visualizing the integrated dataset. The motivation behind integrating all groups was to investigate the collective response of microglia and astrocytes to different apolipoprotein E (APOE) alleles alongside an immune challenge, aiming to gain deeper insights into the interplay between genetic factors and immune activation in these cell populations.

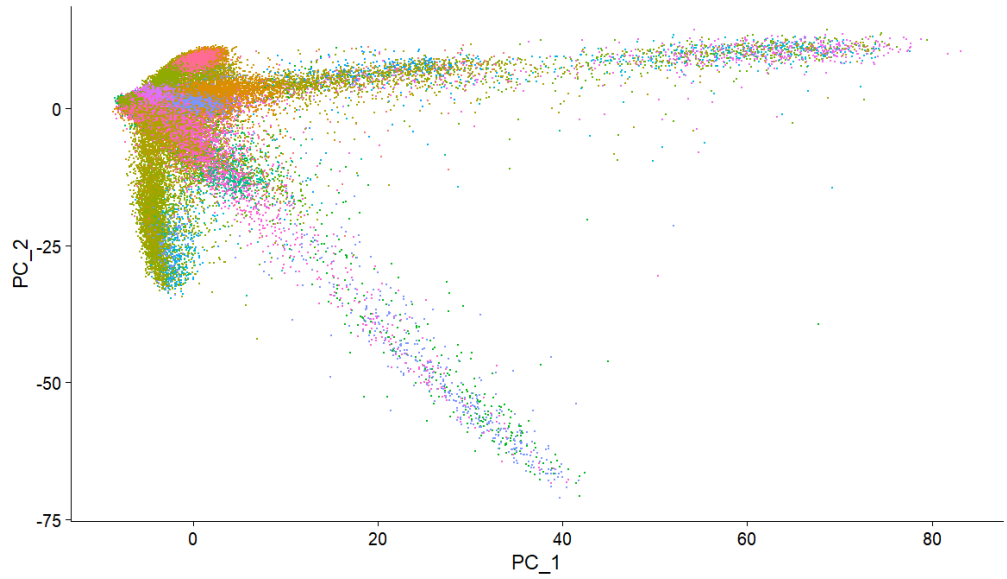

Figure 3: Principal Component Analysis for Merged Groups.

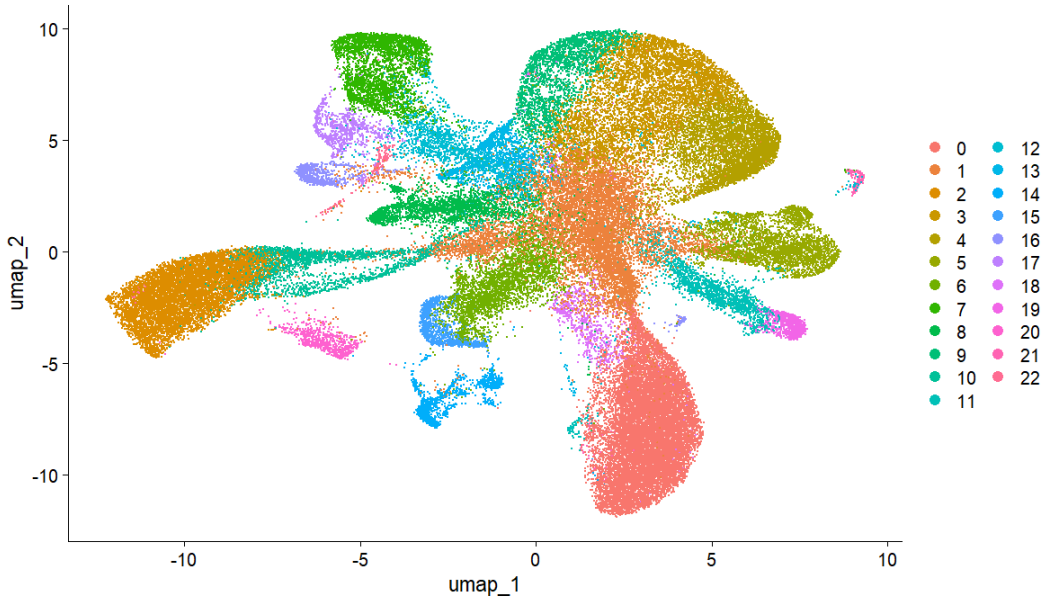

Figure 4: UMAP and Clustering for Merged Groups.

RESULTS AND DISCUSSION

Differentially Expression Analysis

After clustering, each cluster is annotated based on known marker genes or additional analyses, such as differential gene expression, to assign biological labels (e.g., cell types) to the clusters.

Genes that are significantly upregulated or downregulated between different conditions or cell types are identified. This analysis helps identify genes that are associated with specific biological processes, pathways, or cell states, providing insights into the underlying molecular mechanisms.

An analysis was conducted to compare each cluster against the others within the same UMAP in order to identify the top 5 genes expressed in that cluster that would distinguish it from all others. Based on this, a cell identity was assigned to each cluster within the UMAPs. The identification of the clusters was validated by comparing the markers of microglia, astrocytes, and oligodendrocytes directly within the clusters and finding correlations between these signature genes of each cell type with their respective clusters. However, for future work, it may be beneficial to analyze up to 10 differentially expressed genes in each cluster for more robust results. Since each cluster took approximately 20 to 30 minutes to identify just 5 genes, and with some groups having up to 21 clusters and a total of 4 groups, due to project time constraints, a more in-depth analysis could not be conducted. As a result, some of the correlations between genes and clusters are not perfect.

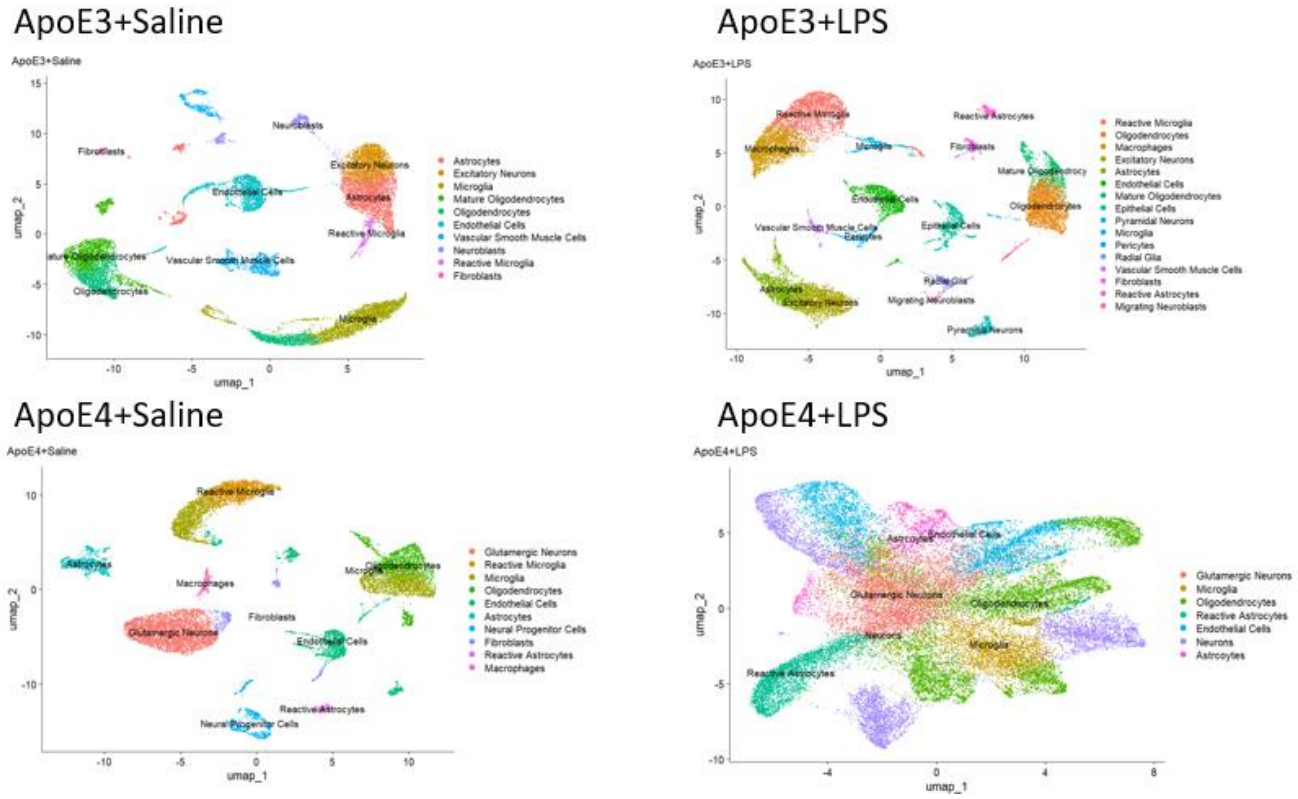

Figure 5: Different Cell Type Classification.

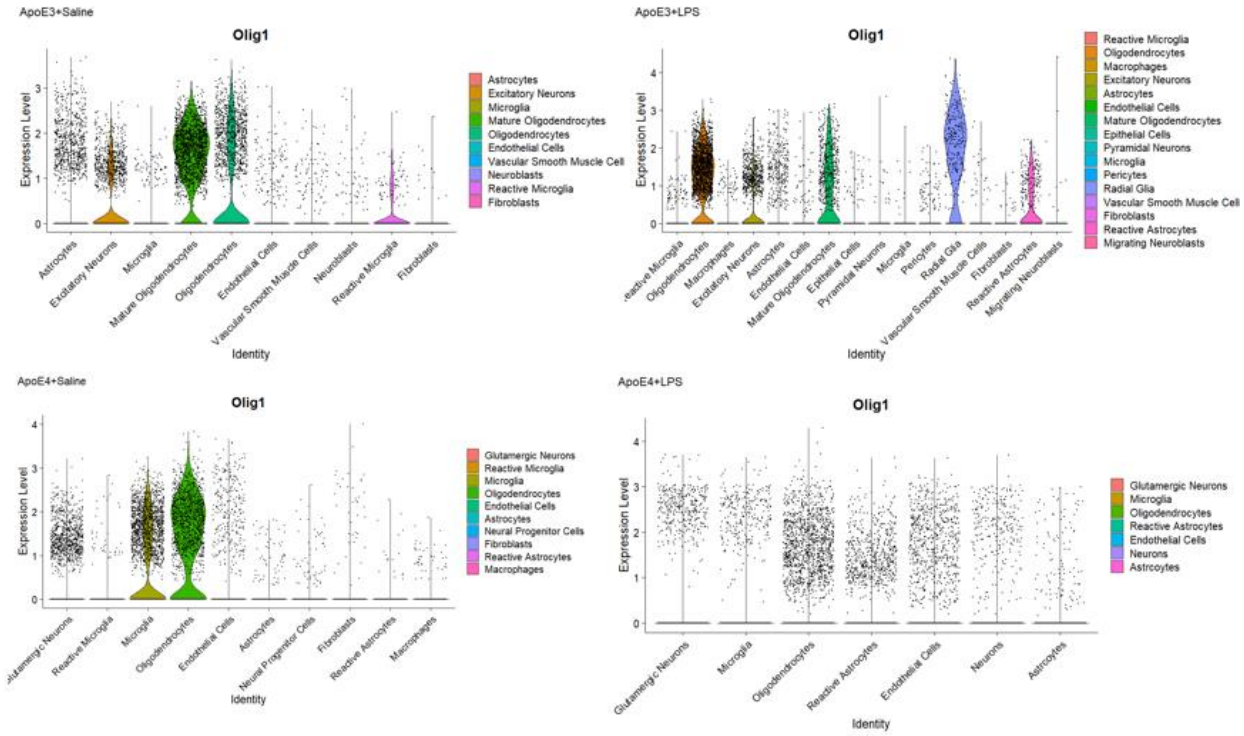

Figure 6: Olig1 (Oligodendrocytes Markers) Expression Through Clusters.

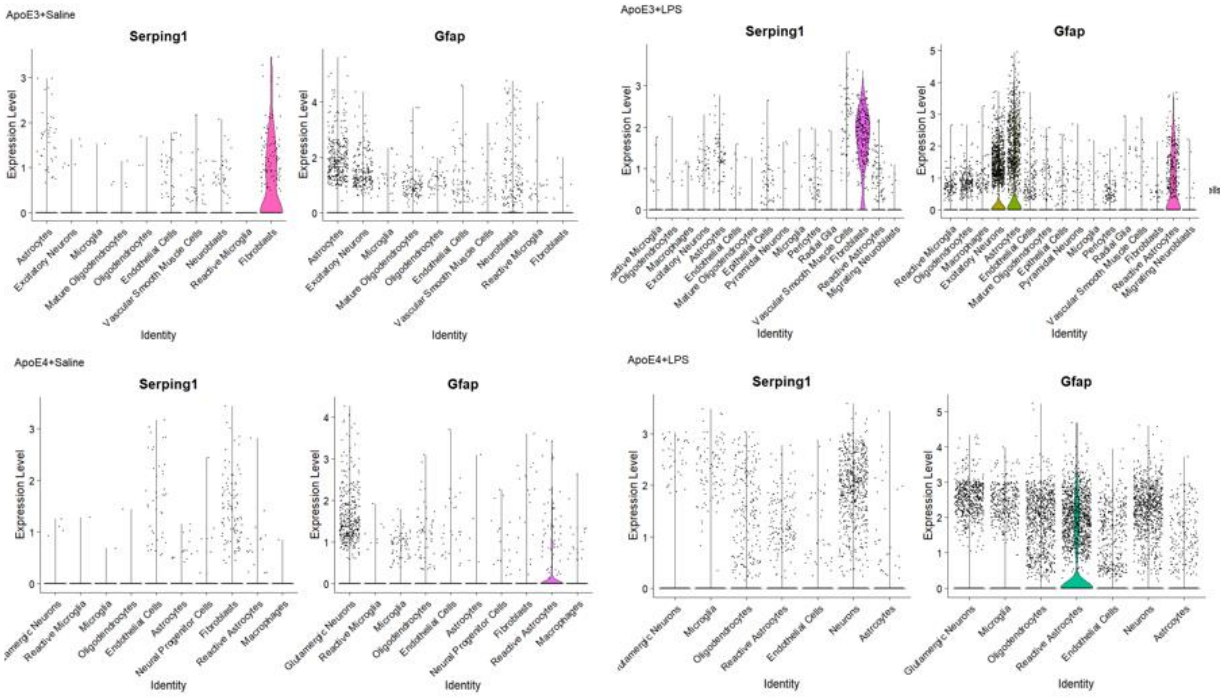

Figure 7: Serping1 and Gfap (Reactive Astrocytes Markers) Expression Through Clusters.

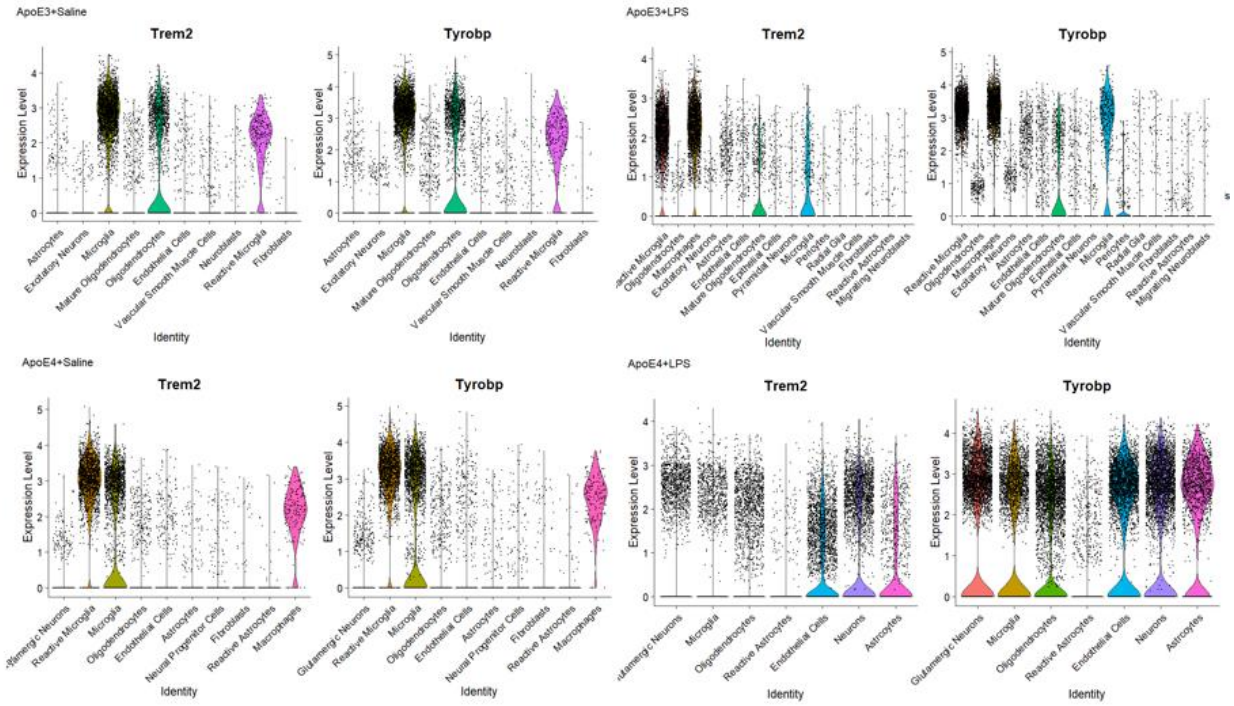

Figure 8: Trem2 and Tyrobp (Reactive Microglia Markers) Expression Through Clusters.

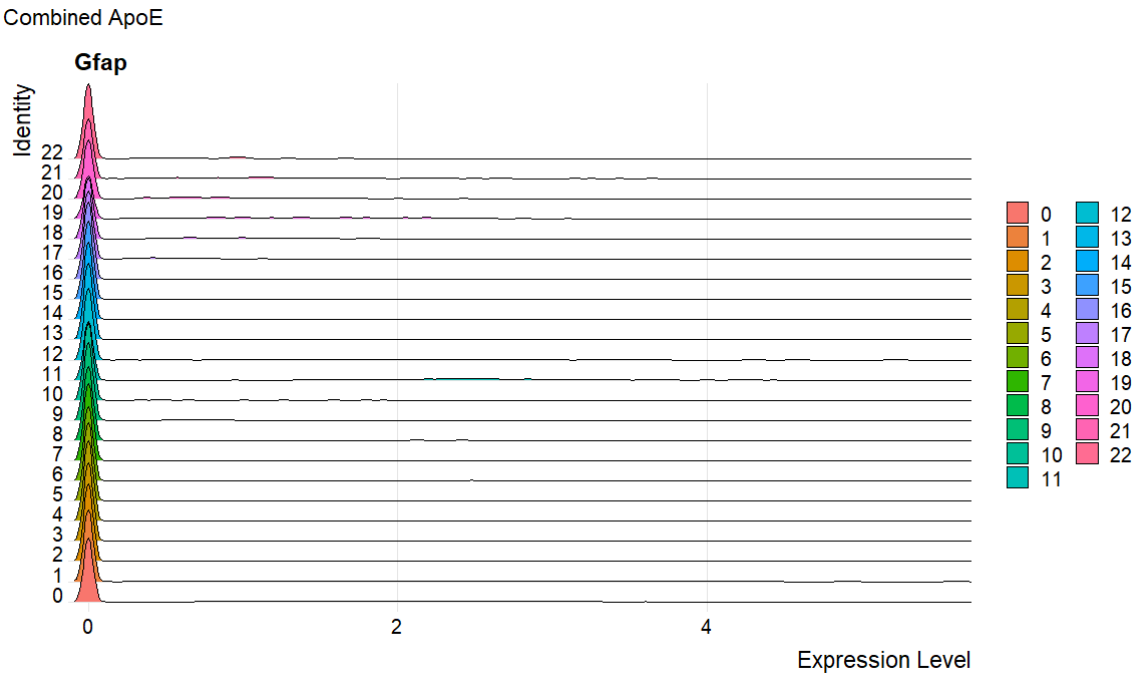

Figure 9: Ridge Plot of Gfap Expression in Merged Seurat Objects

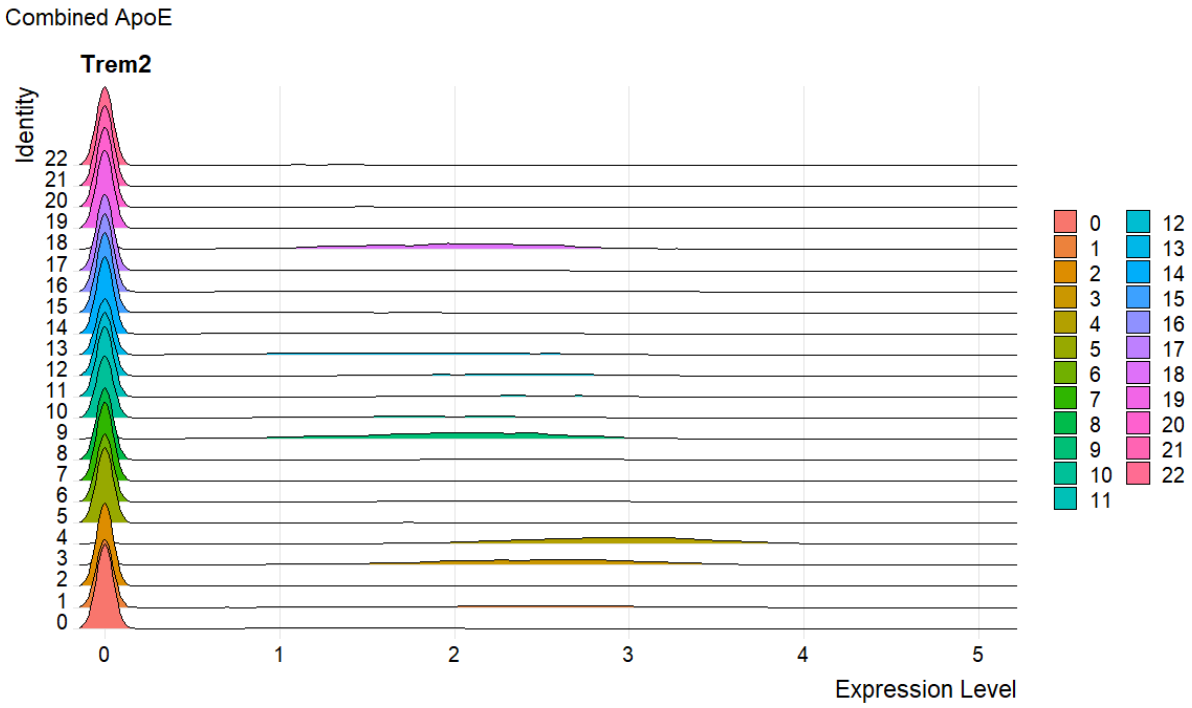

Figure 10: Ridge Plot of Trem2 Expression in Merged Seurat Objects

CONCLUSIONS

By examining the gene expression profiles specific to certain cell types, the correct assignment of cell identity to the majority of clusters across the four groups was confirmed. Additionally, intriguingly, when focusing on the clusters from the ApoE4+LPS group and comparing their results with the others, it is observed that all cells converge with each other, and there is not as clear a distinction as in the other groups. This is likely due to the combination of both factors (ApoE4 and LPS), which may be causing a potential inflammatory reaction throughout the tissue, resulting in different cell types becoming more homogeneous in their expression of inflammatory markers. Furthermore, it appears that this effect is more pronounced in microglia than in astrocytes, as markers related to disease-associated microglia (DAM) are expressed more than those related to reactive astrocytes. Finally, due to time constraints and the aforementioned time required to classify the cell identity of each cluster, this analysis was not performed in the Seurat object with all combined objects. However, by creating a ridge plot of the expression of certain markers for microglia and astrocytes, it is evident that this approach would be more advantageous for a study involving microglia, as changes in the clusters are less pronounced when combining all groups for astrocytes. Conversely, with microglia, the changes in clusters are more apparent when combined.

REFERENCES

Lee, Sangderk, et al. “APOE modulates microglial immunometabolism in response to age, amyloid pathology, and inflammatory challenge.” *Cell Reports*, vol. 42, no. 3, Mar. 2023, p. 112196, https://doi.org/10.1016/j.celrep.2023.112196.

Loulier, Karine. “Faculty opinions recommendation of reactive astrocyte nomenclature, definitions, and future directions.” *Faculty Opinions – Post-Publication Peer Review of the Biomedical Literature*, 10 Mar. 2021, https://doi.org/10.3410/f.739564132.793583692.

Patrick Miller-Rhodes, PhD. “A Guide to Microglia Markers.” *Biocompare*, 10 June 2022, www.biocompare.com/Editorial-Articles/586779-A-Guide-to-Microglial-Markers/#:~:text=These%20microglia%2C%20called%20disease%2Dassociated,among%20other%20genes%2C%20are%20upregulated.

“Seurat - Guided Clustering Tutorial.” • *Seurat*, satijalab.org/seurat/articles/pbmc3k_tutorial. Accessed 2 May 2024.

CODE

```
install.packages('dplyr')

install.packages('Seurat')

install.packages('gridExtra')

install.packages("fs")

install.packages('devtools')

devtools::install_github('immunogenomics/presto')


library(dplyr)

library(Seurat)

library(patchwork)

library(gridExtra)

library(devtools)

library(ggplot2)


options(java.parameters = "-Xmx8g")

options(java.parameters = "-Xmx48g")



# Load the PBMC dataset

ApoE3LPS.data <- Read10X(data.dir = "C:/Users/hzepeda6/Desktop/Single Cell/APOE modulates microglial immunometabolism in response to age, amyloid pathology, and inflammatory challenge/APOExLPS/filtered_gene_bc_matrices/ApoE3LPS/hg19")

ApoE3NaCl.data <- Read10X(data.dir = "C:/Users/hzepeda6/Desktop/Single Cell/APOE modulates microglial immunometabolism in response to age, amyloid pathology, and inflammatory challenge/APOExLPS/filtered_gene_bc_matrices/ApoE3NaCl/hg19")

ApoE4LPS.data <- Read10X(data.dir = "C:/Users/hzepeda6/Desktop/Single Cell/APOE modulates microglial immunometabolism in response to age, amyloid pathology, and inflammatory challenge/APOExLPS/filtered_gene_bc_matrices/ApoE4LPS/hg19")

ApoE4NaCl.data <- Read10X(data.dir = "C:/Users/hzepeda6/Desktop/Single Cell/APOE modulates microglial immunometabolism in response to age, amyloid pathology, and inflammatory challenge/APOExLPS/filtered_gene_bc_matrices/ApoE4NaCl/hg19")
```

```
ApoE3LPS <- CreateSeuratObject(counts = ApoE3LPS.data, project = "ApoE3LPS", min.cells = 3, min.features = 200)

ApoE3NaCl <- CreateSeuratObject(counts = ApoE3NaCl.data, project = "ApoE3NaCl", min.cells = 3, min.features = 200)

ApoE4LPS <- CreateSeuratObject(counts = ApoE4LPS.data, project = "ApoE4LPS", min.cells = 3, min.features = 200)

ApoE4NaCl <- CreateSeuratObject(counts = ApoE4NaCl.data, project = "ApoE4NaCl", min.cells = 3, min.features = 200)


# The [[ operator can add columns to object metadata. This is a great place to stash QC stats

ApoE3LPS[["percent.mt"]] <- PercentageFeatureSet(ApoE3LPS, pattern = "^MT-")

ApoE3NaCl[["percent.mt"]] <- PercentageFeatureSet(ApoE3NaCl, pattern = "^MT-")

ApoE4LPS[["percent.mt"]] <- PercentageFeatureSet(ApoE4LPS, pattern = "^MT-")

ApoE4NaCl[["percent.mt"]] <- PercentageFeatureSet(ApoE4NaCl, pattern = "^MT-")


# Visualize QC metrics as a violin plot

#VlnPlotApoE3LPS <- VlnPlot(ApoE3LPS, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

#VlnPlotApoE3NaCl <- VlnPlot(ApoE3NaCl, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

#VlnPlotApoE4LPS <- VlnPlot(ApoE4LPS, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

#VlnPlotApoE4NaCl <- VlnPlot(ApoE4NaCl, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

#grid.arrange(VlnPlotApoE3LPS, VlnPlotApoE3NaCl, VlnPlotApoE4LPS, VlnPlotApoE4NaCl, nrow = 8, ncol = 8)

#VlnPlotApoE3LPS+VlnPlotApoE3NaCl+VlnPlotApoE4LPS+VlnPlotApoE4NaCl


VlnPlot(ApoE3LPS, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

VlnPlot(ApoE3NaCl, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

VlnPlot(ApoE4LPS, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

VlnPlot(ApoE4NaCl, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)
```

```
# FeatureScatter is typically used to visualize feature-feature relationships, but can be used
# for anything calculated by the object, i.e. columns in object metadata, PC scores etc.

plot1 <- FeatureScatter(ApoE3LPS, feature1 = "nCount_RNA", feature2 = "percent.mt")
plot2 <- FeatureScatter(ApoE3LPS, feature1 = "nCount_RNA", feature2 = "nFeature_RNA")
plot1 + plot2

ApoE3LPS <- NormalizeData(ApoE3LPS, normalization.method = "LogNormalize", scale.factor = 10000)
ApoE3LPS <- NormalizeData(ApoE3LPS)
ApoE3LPS <- FindVariableFeatures(ApoE3LPS, selection.method = "vst", nfeatures = 2000)

ApoE3NaCl <- NormalizeData(ApoE3NaCl, normalization.method = "LogNormalize", scale.factor = 10000)
ApoE3NaCl <- NormalizeData(ApoE3NaCl)
ApoE3NaCl <- FindVariableFeatures(ApoE3NaCl, selection.method = "vst", nfeatures = 2000)

ApoE4LPS <- NormalizeData(ApoE4LPS, normalization.method = "LogNormalize", scale.factor = 10000)
ApoE4LPS <- NormalizeData(ApoE4LPS)
ApoE4LPS <- FindVariableFeatures(ApoE4LPS, selection.method = "vst", nfeatures = 2000)

ApoE4NaCl <- NormalizeData(ApoE4NaCl, normalization.method = "LogNormalize", scale.factor = 10000)
ApoE4NaCl <- NormalizeData(ApoE4NaCl)
ApoE4NaCl <- FindVariableFeatures(ApoE4NaCl, selection.method = "vst", nfeatures = 2000)

# Identify the 10 most highly variable genes
top10 <- head(VariableFeatures(ApoE3LPS), 10)

# plot variable features with and without labels
plot1 <- VariableFeaturePlot(ApoE3LPS)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
plot1 + plot2
```

```
top10 <- head(VariableFeatures(ApoE3NaCl), 10)

# plot variable features with and without labels
plot1 <- VariableFeaturePlot(ApoE3NaCl)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
plot1 + plot2

top10 <- head(VariableFeatures(ApoE4LPS), 10)

# plot variable features with and without labels
plot1 <- VariableFeaturePlot(ApoE4LPS)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
plot1 + plot2

top10 <- head(VariableFeatures(ApoE4NaCl), 10)

# plot variable features with and without labels
plot1 <- VariableFeaturePlot(ApoE4NaCl)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
plot1 + plot2

all.genes <- rownames(ApoE3LPS)
ApoE3LPS <- ScaleData(ApoE3LPS, features = all.genes)

ApoE3LPS <- RunPCA(ApoE3LPS, features = VariableFeatures(object = ApoE3LPS))

# Examine and visualize PCA results a few different ways
print(ApoE3LPS[["pca"]], dims = 1:5, nfeatures = 5)
```

```
VizDimLoadings(ApoE3LPS, dims = 1:2, reduction = "pca")

DimPlot(ApoE3LPS, reduction = "pca") + NoLegend()
DimPlotApoE3LPS <- DimPlot(ApoE3LPS, reduction = "pca") + NoLegend()

DimHeatmap(ApoE3LPS, dims = 1:15, cells = 500, balanced = TRUE)
####################
all.genes <- rownames(ApoE3NaCl)
ApoE3NaCl <- ScaleData(ApoE3NaCl, features = all.genes)

ApoE3NaCl <- RunPCA(ApoE3NaCl, features = VariableFeatures(object = ApoE3NaCl))

# Examine and visualize PCA results a few different ways
print(ApoE3NaCl[["pca"]], dims = 1:5, nfeatures = 5)

VizDimLoadings(ApoE3NaCl, dims = 1:2, reduction = "pca")

DimPlot(ApoE3NaCl, reduction = "pca") + NoLegend()
DimPlotApoE3NaCl <- DimPlot(ApoE3NaCl, reduction = "pca") + NoLegend()

DimHeatmap(ApoE3NaCl, dims = 1:15, cells = 500, balanced = TRUE)
###################
all.genes <- rownames(ApoE4LPS)
ApoE4LPS <- ScaleData(ApoE4LPS, features = all.genes)

ApoE4LPS <- RunPCA(ApoE4LPS, features = VariableFeatures(object = ApoE4LPS))

# Examine and visualize PCA results a few different ways
print(ApoE4LPS[["pca"]], dims = 1:5, nfeatures = 5)
```

```
VizDimLoadings(ApoE4LPS, dims = 1:2, reduction = "pca")

DimPlot(ApoE4LPS, reduction = "pca") + NoLegend()
DimPlotApoE4LPS <- DimPlot(ApoE4LPS, reduction = "pca") + NoLegend()

DimHeatmap(ApoE4LPS, dims = 1:15, cells = 500, balanced = TRUE)
##################
all.genes <- rownames(ApoE4NaCl)
ApoE4NaCl <- ScaleData(ApoE4NaCl, features = all.genes)

ApoE4NaCl <- RunPCA(ApoE4NaCl, features = VariableFeatures(object = ApoE4NaCl))

# Examine and visualize PCA results a few different ways
print(ApoE4NaCl[["pca"]], dims = 1:5, nfeatures = 5)

VizDimLoadings(ApoE4NaCl, dims = 1:2, reduction = "pca")

DimPlot(ApoE4NaCl, reduction = "pca") + NoLegend()

DimPlotApoE4NaCl <- DimPlot(ApoE4NaCl, reduction = "pca") + NoLegend()

DimPlotApoE3LPS+DimPlotApoE3NaCl+DimPlotApoE4LPS+DimPlotApoE4NaCl

DimHeatmap(ApoE4NaCl, dims = 1:15, cells = 500, balanced = TRUE)
################################################################
ApoE3LPS <- FindNeighbors(ApoE3LPS, dims = 1:10)
ApoE3LPS <- FindClusters(ApoE3LPS, resolution = 0.5)

# Look at cluster IDs of the first 5 cells
head(Idents(ApoE3LPS), 5)
```

```
ApoE3LPS <- RunUMAP(ApoE3LPS, dims = 1:10)

DimPlotApoE3LPS <-DimPlot(ApoE3LPS, reduction = "umap")
DimPlotApoE3LPS
##################
ApoE3NaCl <- FindNeighbors(ApoE3NaCl, dims = 1:10)
ApoE3NaCl <- FindClusters(ApoE3NaCl, resolution = 0.5)

# Look at cluster IDs of the first 5 cells
head(Idents(ApoE3NaCl), 5)

ApoE3NaCl <- RunUMAP(ApoE3NaCl, dims = 1:10)

DimPlotApoE3NaCl<-DimPlot(ApoE3NaCl, reduction = "umap")
DimPlotApoE3NaCl
################
ApoE4LPS <- FindNeighbors(ApoE4LPS, dims = 1:10)
ApoE4LPS <- FindClusters(ApoE4LPS, resolution = 0.5)

# Look at cluster IDs of the first 5 cells
head(Idents(ApoE4LPS), 5)

ApoE4LPS <- RunUMAP(ApoE4LPS, dims = 1:10)

DimPlotApoE4LPS<-DimPlot(ApoE4LPS, reduction = "umap")
DimPlotApoE4LPS
################
ApoE4NaCl <- FindNeighbors(ApoE4NaCl, dims = 1:10)
ApoE4NaCl <- FindClusters(ApoE4NaCl, resolution = 0.5)
```

```
# Look at cluster IDs of the first 5 cells
head(Idents(ApoE4NaCl), 5)
################################### Different Groups UMAPS
ApoE4NaCl <- RunUMAP(ApoE4NaCl, dims = 1:10)
DimPlotApoE4NaCl <- DimPlot(ApoE4NaCl, reduction = "umap")
DimPlotApoE4NaCl
#DimPlotApoE3LPS+DimPlotApoE3NaCl+DimPlotApoE4LPS+DimPlotApoE4NaCl
(DimPlotApoE3LPS+DimPlotApoE3NaCl+DimPlotApoE4LPS+DimPlotApoE4NaCl) & NoLegend()
#grid.arrange(DimPlotApoE3LPS, DimPlotApoE3NaCl, DimPlotApoE4LPS, DimPlotApoE4NaCl, nrow = 2, ncol = 2,top = "UMAPs", grobs=gs)
DimPlotApoE3LPS + plot_annotation(title = 'ApoE3+LPS')
DimPlotApoE3NaCl + plot_annotation(title = 'ApoE3+Saline')
DimPlotApoE4LPS + plot_annotation(title = 'ApoE4+LPS')
DimPlotApoE4NaCl + plot_annotation(title = 'ApoE4+Saline')
################################### Violin Plots to visualize microglia markers in the different groups
# Define the cluster identifiers
cluster_ids <- c(0, 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,16,17,18)

# Initialize an empty list to store cluster markers
cluster_markers_list <- list()

# Iterate over each cluster identifier
for (cluster_id in cluster_ids) {
  # Find markers for the current cluster
  cluster_markers <- FindMarkers(ApoE3LPS, ident.1 = cluster_id, ident.2 = NULL)

  # Store the cluster markers in the list
  cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]] <- cluster_markers
}
```

```
# Print the head of cluster markers for each cluster
for (cluster_id in cluster_ids) {
  # Extract the cluster markers from the list
  cluster_markers <- cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]]

  # Print the head of cluster markers
  cat("Cluster", cluster_id, "markers:\n")
  print(head(cluster_markers, n = 5))
  cat("\n")
}

# Define the cluster identifiers
cluster_ids <- c(0, 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21)

# Initialize an empty list to store cluster markers
cluster_markers_list <- list()

# Iterate over each cluster identifier
for (cluster_id in cluster_ids) {
  # Find markers for the current cluster
  cluster_markers <- FindMarkers(ApoE3NaCl, ident.1 = cluster_id, ident.2 = NULL)

  # Store the cluster markers in the list
  cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]] <- cluster_markers
}

# Print the head of cluster markers for each cluster
for (cluster_id in cluster_ids) {
  # Extract the cluster markers from the list
  cluster_markers <- cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]]
```

```
  # Print the head of cluster markers
  cat("Cluster", cluster_id, "markers:\n")
  print(head(cluster_markers, n = 5))
  cat("\n")
}

# Define the cluster identifiers
cluster_ids <- c(0, 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15)

# Initialize an empty list to store cluster markers
cluster_markers_list <- list()

# Iterate over each cluster identifier
for (cluster_id in cluster_ids) {
  # Find markers for the current cluster
  cluster_markers <- FindMarkers(ApoE4LPS, ident.1 = cluster_id, ident.2 = NULL)

  # Store the cluster markers in the list
  cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]] <- cluster_markers
}

# Print the head of cluster markers for each cluster
for (cluster_id in cluster_ids) {
  # Extract the cluster markers from the list
  cluster_markers <- cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]]

  # Print the head of cluster markers
  cat("Cluster", cluster_id, "markers:\n")
  print(head(cluster_markers, n = 5))
```

```
  cat("\n")
}

# Define the cluster identifiers
cluster_ids <- c(0, 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)

# Initialize an empty list to store cluster markers
cluster_markers_list <- list()

# Iterate over each cluster identifier
for (cluster_id in cluster_ids) {
  # Find markers for the current cluster
  cluster_markers <- FindMarkers(ApoE4NaCl, ident.1 = cluster_id, ident.2 = NULL)

  # Store the cluster markers in the list
  cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]] <- cluster_markers
}

# Print the head of cluster markers for each cluster
for (cluster_id in cluster_ids) {
  # Extract the cluster markers from the list
  cluster_markers <- cluster_markers_list[[paste0("cluster", cluster_id, ".markers")]]

  # Print the head of cluster markers
  cat("Cluster", cluster_id, "markers:\n")
  print(head(cluster_markers, n = 5))
  cat("\n")
}

############################## ApoE3LPS
```

```
new.cluster.idsApoe3LPS <- c("Reactive Microglia", "Oligodendrocytes", "Macrophages",
                             "Excitatory Neurons", "Astrocytes", "Endothelial Cells",
                             "Mature Oligodendrocytes", "Epithelial Cells",
                             "Pyramidal Neurons", "Microglia","Pericytes","Radial Glia",
                             "Vascular Smooth Muscle Cells","Fibroblasts", "Reactive Astrocytes",
                             "Reactive Astrocytes","Migrating Neuroblasts", "Reactive Microglia",
                             "Endothelial Cells")

# Assign cluster names to Seurat object
names(new.cluster.idsApoe3LPS) <- levels(ApoE3LPS)

# Rename cluster identities
ApoE3LPS <- RenameIdents(ApoE3LPS, new.cluster.idsApoe3LPS)

# Plot UMAP with cluster names
DimPlot(ApoE3LPS, reduction = "umap", label = TRUE, pt.size = 0.5) +  plot_annotation(title = 'ApoE3+LPS')

# Visualize gene expression using FeaturePlot
FeaturePlot(ApoE3LPS, features = c("Serping1", "Gfap")) +  plot_annotation(title = 'ApoE3+LPS')

FeaturePlot(ApoE3LPS, features = c("Ctsb","Ctsd","Ctsl")) +  plot_annotation(title = 'ApoE3+LPS')

FeaturePlot(ApoE3LPS, features = c("Plaur","Mmp2","Mmp13")) +  plot_annotation(title = 'ApoE3+LPS')
FeaturePlot(ApoE3LPS, features = c("Axin2","Nes","Ctnnb1")) +  plot_annotation(title = 'ApoE3+LPS')

VlnPlot(ApoE3LPS, features = c("Serping1", "Gfap")) +  plot_annotation(title = 'ApoE3+LPS')
VlnPlot(ApoE3LPS, features = c("Ctsb", "Ctsl")) +  plot_annotation(title = 'ApoE3+LPS')
VlnPlot(ApoE3LPS, features = c("Trem2","Tyrobp")) +  plot_annotation(title = 'ApoE3+LPS')
VlnPlot(ApoE3LPS, features = c("Aebp1","Wwtr1")) +  plot_annotation(title = 'ApoE3+LPS')
VlnPlot(ApoE3LPS, features = c("Phyhd1","Dst","Rasl12")) +  plot_annotation(title = 'ApoE3+LPS')
```

```
VlnPlot(ApoE3LPS, features = c("Olig1")) +  plot_annotation(title = 'ApoE3+LPS')

##############################################Apoe3+Saline
new.cluster.idsApoE3NaCl <- c("Astrocytes", "Excitatory Neurons", "Microglia",
                              "Mature Oligodendrocytes", "Oligodendrocytes", "Endothelial Cells",
                              "Oligodendrocytes", "Microglia", "Vascular Smooth Muscle Cells",
                              "Vascular Smooth Muscle Cells","Neuroblasts","Endothelial Cells",
                              "Astrocytes","Microglia","Reactive Microglia","Mature Oligodendrocytes",
                              "Endothelial Cells","Mature Oligodendrocytes","Astrocytes","Fibroblasts",
                              "Neuroblasts","Endothelial Cells")

# Assign cluster names to Seurat object
names(new.cluster.idsApoE3NaCl) <- levels(ApoE3NaCl)

# Rename cluster identities
ApoE3NaCl <- RenameIdents(ApoE3NaCl, new.cluster.idsApoE3NaCl)

# Plot UMAP with cluster names
DimPlot(ApoE3NaCl, reduction = "umap", label = TRUE, pt.size = 0.5) +  plot_annotation(title = 'ApoE3+Saline')

# Visualize gene expression using FeaturePlot
FeaturePlot(ApoE3NaCl, features = c("Serping1", "Gfap", "Ggta")) +  plot_annotation(title = 'ApoE3+Saline')
FeaturePlot(ApoE3NaCl, features = c("Ctsb","Ctsd","Ctsl")) +  plot_annotation(title = 'ApoE3+Saline')

FeaturePlot(ApoE3NaCl, features = c("Plaur","Mmp2","Mmp13")) +  plot_annotation(title = 'ApoE3+Saline')
FeaturePlot(ApoE3NaCl, features = c("Axin2","Nes","Ctnnb1")) +  plot_annotation(title = 'ApoE3+Saline')

# Get the gene expression matrix
expression_matrix <- GetAssayData(ApoE3NaCl, assay = "RNA")
```

```
# Extract gene expression values for specific genes
gene_expression <- expression_matrix[c("Serping1", "Gfap"), ]

VlnPlot(ApoE3NaCl, features = c("Serping1", "Gfap")) +  plot_annotation(title = 'ApoE3+Saline')
VlnPlot(ApoE3NaCl, features = c("Ctbs", "Ctsl")) +  plot_annotation(title = 'ApoE3+Saline')
VlnPlot(ApoE3NaCl, features = c("Trem2","Tyrobp")) +  plot_annotation(title = 'ApoE3+Saline')
VlnPlot(ApoE3NaCl, features = c("Aebp1","Wwtr1")) +  plot_annotation(title = 'ApoE3+Saline')
VlnPlot(ApoE3NaCl, features = c("Phyhd1","Dst","Rasl12")) +  plot_annotation(title = 'ApoE3+Saline')
VlnPlot(ApoE3NaCl, features = c("Olig1")) +  plot_annotation(title = 'ApoE3+Saline')

#######################################Apoe4+LPS
new.cluster.idsApoE4LPS <- c("Glutamergic Neurons","Microglia","Oligodendrocytes",
                  "Reactive Astrocytes","Oligodendrocytes","Endothelial Cells",
                  "Neurons","Neurons","Endothelial Cells",
                  "Astrcoytes","Neurons","Oligodendrocytes",
                  "Oligodendrocytes","Oligodendrocytes","Microglia",
                  "Reactive Astrocytes","Neural Progenitor Cells","Monocytes",
                  "Endothelial Cells","Endothelial Cells")

# Assign cluster names to Seurat object
names(new.cluster.idsApoE4LPS) <- levels(ApoE4LPS)

# Rename cluster identities
ApoE4LPS <- RenameIdents(ApoE4LPS, new.cluster.idsApoE4LPS)

# Plot UMAP with cluster names
DimPlot(ApoE4LPS, reduction = "umap", label = TRUE, pt.size = 0.5) +  plot_annotation(title = 'ApoE4+LPS')

# Visualize gene expression using FeaturePlot
FeaturePlot(ApoE4LPS, features = c("Serping1", "Gfap", "Ggta")) +  plot_annotation(title = 'ApoE4+LPS')
```

```
FeaturePlot(ApoE4LPS, features = c("Ctsb","Ctsd","Ctsl")) +  plot_annotation(title = 'ApoE4+LPS')

FeaturePlot(ApoE4LPS, features = c("Plaur","Mmp2","Mmp13")) +  plot_annotation(title = 'ApoE4+LPS')
FeaturePlot(ApoE4LPS, features = c("Axin2","Nes","Ctnnb1")) +  plot_annotation(title = 'ApoE4+LPS')

VlnPlot(ApoE4LPS, features = c("Serping1", "Gfap")) +  plot_annotation(title = 'ApoE4+LPS')
VlnPlot(ApoE4LPS, features = c("Ctsb", "Ctsl")) +  plot_annotation(title = 'ApoE4+LPS')
VlnPlot(ApoE4LPS, features = c("Trem2","Tyrobp")) +  plot_annotation(title = 'ApoE4+LPS')
VlnPlot(ApoE4LPS, features = c("Aebp1","Wwtr1")) +  plot_annotation(title = 'ApoE4+LPS')
VlnPlot(ApoE4LPS, features = c("Phyhd1","Dst","Rasl12")) +  plot_annotation(title = 'ApoE4+LPS')
VlnPlot(ApoE4LPS, features = c("Olig1")) +  plot_annotation(title = 'ApoE4+LPS')

####################################Apoe4+Saline
new.cluster.idsApoE4NaCl <- c("Glutamergic Neurons", "Reactive Microglia", "Microglia",
                              "Microglia", "Oligodendrocytes", "Endothelial Cells",
                              "Astrocytes", "Neural Progenitor Cells", "Endothelial Cells",
                              "Fibroblasts", "Endothelial Cells", "Reactive Astrocytes",
                              "Oligodendrocytes", "Oligodendrocytes", "Macrophages",
                              "Fibroblasts", "Neural Progenitor Cells", "Endothelial Cells",
                              "Fibroblasts", "Endothelial Cells")

# Assign cluster names to Seurat object
names(new.cluster.idsApoE4NaCl) <- levels(ApoE4NaCl)

# Rename cluster identities
ApoE4NaCl <- RenameIdents(ApoE4NaCl, new.cluster.idsApoE4NaCl)

# Plot UMAP with cluster names
DimPlot(ApoE4NaCl, reduction = "umap", label = TRUE, pt.size = 0.5) +  plot_annotation(title = 'ApoE4+Saline')
```

```
FeaturePlot(ApoE4NaCl, features = c("Serping1", "Gfap", "Ggta"))  +  plot_annotation(title = 'ApoE4+Saline')
FeaturePlot(ApoE4NaCl, features = c("Ctsb","Ctsd","Ctsl")) +  plot_annotation(title = 'ApoE4+NaCl')
FeaturePlot(ApoE4NaCl, features = c("Plaur","Mmp2","Mmp13")) +  plot_annotation(title = 'ApoE4+Saline')
FeaturePlot(ApoE4NaCl, features = c("Axin2","Nes","Ctnnb1")) +  plot_annotation(title = 'ApoE4+Saline')

VlnPlot(ApoE4NaCl, features = c("Serping1", "Gfap")) +  plot_annotation(title = 'ApoE4+Saline')
VlnPlot(ApoE4NaCl, features = c("Ctsb", "Ctsl")) +  plot_annotation(title = 'ApoE4+Saline')
VlnPlot(ApoE4NaCl, features = c("Trem2","Tyrobp")) +  plot_annotation(title = 'ApoE4+Saline')
VlnPlot(ApoE4NaCl, features = c("Aebp1","Wwtr1")) +  plot_annotation(title = 'ApoE4+Saline')
VlnPlot(ApoE4NaCl, features = c("Phyhd1","Dst","Rasl12")) +  plot_annotation(title = 'ApoE4+Saline')

VlnPlot(ApoE4NaCl, features = c("Olig1")) +  plot_annotation(title = 'ApoE4+Saline')

################################################
plots.dir.path <- list.files(tempdir(), pattern="rs-graphics", full.names = TRUE);
plots.png.paths <- list.files(plots.dir.path, pattern=".png", full.names = TRUE)

RidgePlot(ApoE4LPS, features = c("Gfap"), ncol = 1)+  plot_annotation(title = 'ApoE4+LPS')
RidgePlot(ApoE3LPS, features = c("Gfap"), ncol = 1)+  plot_annotation(title = 'ApoE3+LPS')
RidgePlot(ApoE3NaCl, features = c("Gfap"), ncol = 1)+  plot_annotation(title = 'ApoE3+Saline')
RidgePlot(ApoE4NaCl, features = c("Gfap"), ncol = 1)+  plot_annotation(title = 'ApoE4+Saline')

RidgePlot(ApoE4LPS, features = c("Wwtr1"), ncol = 1)+  plot_annotation(title = 'ApoE4+LPS')
RidgePlot(ApoE3LPS, features = c("Wwtr1"), ncol = 1)+  plot_annotation(title = 'ApoE3+LPS')
RidgePlot(ApoE3NaCl, features = c("Wwtr1"), ncol = 1)+  plot_annotation(title = 'ApoE3+Saline')
RidgePlot(ApoE4NaCl, features = c("Wwtr1"), ncol = 1)+  plot_annotation(title = 'ApoE4+Saline')

RidgePlot(ApoE4LPS, features = c("Trem2"), ncol = 1)+  plot_annotation(title = 'ApoE4+LPS')
RidgePlot(ApoE3LPS, features = c("Trem2"), ncol = 1)+  plot_annotation(title = 'ApoE3+LPS')
RidgePlot(ApoE3NaCl, features = c("Trem2"), ncol = 1)+  plot_annotation(title = 'ApoE3+Saline')
```

```
RidgePlot(ApoE4NaCl, features = c("Trem2"), ncol = 1)+  plot_annotation(title = 'ApoE4+Saline')

RidgePlot(ApoE4LPS, features = c("Ctsb"), ncol = 2)

##################################### Combined Groups

pbmc.combinedApoE3 <- merge(ApoE3NaCl, y = ApoE3LPS, add.cell.ids = c("15251", "15927"), project = "PBMC12K")

pbmc.combinedApoE3


pbmc.combinedApoE4 <- merge(ApoE4NaCl, y = ApoE4LPS, add.cell.ids = c("13911", "29259"), project = "PBMC12K")

pbmc.combinedApoE4


pbmc.combinedApoE <- merge(pbmc.combinedApoE3, y = pbmc.combinedApoE4, add.cell.ids = c("31178", "43170"), project = "PBMC12K")

pbmc.combinedApoE

#####################################################

# The [[ operator can add columns to object metadata. This is a great place to stash QC stats

pbmc.combinedApoE[["percent.mt"]] <- PercentageFeatureSet(pbmc.combinedApoE, pattern = "^MT-")


# Visualize QC metrics as a violin plot

VlnPlot(pbmc.combinedApoE, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)


# FeatureScatter is typically used to visualize feature-feature relationships, but can be used

# for anything calculated by the object, i.e. columns in object metadata, PC scores etc.


plot1 <- FeatureScatter(pbmc.combinedApoE, feature1 = "nCount_RNA", feature2 = "percent.mt")

plot2 <- FeatureScatter(pbmc.combinedApoE, feature1 = "nCount_RNA", feature2 = "nFeature_RNA")

plot1 + plot2


pbmc.combinedApoE <- NormalizeData(pbmc.combinedApoE, normalization.method = "LogNormalize", scale.factor = 10000)

pbmc.combinedApoE <- NormalizeData(pbmc.combinedApoE)
```

```
pbmc.combinedApoE <- FindVariableFeatures(pbmc.combinedApoE, selection.method = "vst", nfeatures = 2000)

# Identify the 10 most highly variable genesz
top10 <- head(VariableFeatures(pbmc.combinedApoE), 10)

# plot variable features with and without labels
plot1 <- VariableFeaturePlot(pbmc.combinedApoE)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
plot1 + plot2

all.genes <- rownames(pbmc.combinedApoE)
pbmc.combinedApoE <- ScaleData(pbmc.combinedApoE, features = all.genes)

pbmc.combinedApoE <- RunPCA(pbmc.combinedApoE, features = VariableFeatures(object =
pbmc.combinedApoE))

# Examine and visualize PCA results a few different ways
print(pbmc.combinedApoE[["pca"]], dims = 1:5, nfeatures = 5)

VizDimLoadings(pbmc.combinedApoE, dims = 1:2, reduction = "pca")

DimPlot(pbmc.combinedApoE, reduction = "pca") + NoLegend()

DimHeatmap(pbmc.combinedApoE, dims = 1:15, cells = 500, balanced = TRUE)

pbmc.combinedApoE <- FindNeighbors(pbmc.combinedApoE, dims = 1:10)
pbmc.combinedApoE <- FindClusters(pbmc.combinedApoE, resolution = 0.5)

# Look at cluster IDs of the first 5 cells
head(Idents(pbmc.combinedApoE), 5)
```

```
pbmc.combinedApoE <- RunUMAP(pbmc.combinedApoE, dims = 1:10)

DimPlot(pbmc.combinedApoE, reduction = "umap")

pbmc.combinedApoE <- JoinLayers(pbmc.combinedApoE)

RidgePlot(pbmc.combinedApoE, features = c("Gfap"), ncol = 1)+  plot_annotation(title = 'Combined ApoE')

RidgePlot(pbmc.combinedApoE, features = c("Wwtr1"), ncol = 1)+  plot_annotation(title = 'Combined ApoE')

RidgePlot(pbmc.combinedApoE, features = c("Aebp1"), ncol = 1)+  plot_annotation(title = 'Combined ApoE')

RidgePlot(pbmc.combinedApoE, features = c("Trem2"), ncol = 1)+  plot_annotation(title = 'Combined ApoE')

file.copy(from=plots.png.paths, to="C:/Users/hzepeda6/Desktop/Single Cell/APOE modulates microglial immunometabolism in response to age, amyloid pathology, and inflammatory challenge/New New Plots")
```