Single cell analysis of iPSC-derived midbrain organoids

## Introduction

The following script was used for analysis of gene corrected (GC) versus GBA1 mutant (MUT) midbrain organoids. The purpose was to combine, filter, integrate, and identify clusters and differentially expressed genes sets.

This script is based on the Seurat tutorials

* <https://satijalab.org/seurat/articles/integration_introduction.html>
* <https://ucdavis-bioinformatics-training.github.io/2019-single-cell-RNA-sequencing-Workshop-UCD_UCSF/scrnaseq_analysis/scRNA_Workshop-PART1.html>

## Part 1. Data preparation:

# Install and load the libraries
library(Seurat)
library(patchwork)
library(ggplot2)
library(cowplot)
library(magrittr)
library(tidyverse)

# Load and combine 10x Runs
setwd("D:.../analysis\_results")

GC1.data <- Read10X\_h5("1GC\_filtered\_feature\_bc\_matrix.h5")
GC1 <- CreateSeuratObject(counts = GC1.data)
GC1

MUT1.data <- Read10X\_h5("1MUT\_filtered\_feature\_bc\_matrix.h5")
MUT1 <- CreateSeuratObject(counts = MUT1.data)
MUT1

GC2.data <- Read10X\_h5("2GC\_filtered\_feature\_bc\_matrix.h5")
GC2 <- CreateSeuratObject(counts = GC2.data)
GC2

MUT2.data <- Read10X\_h5("2MUT\_filtered\_feature\_bc\_matrix.h5")
MUT2 <- CreateSeuratObject(counts = MUT2.data)
MUT2

GC3.data <- Read10X\_h5("3GC\_filtered\_feature\_bc\_matrix.h5")
GC3 <- CreateSeuratObject(counts = GC3.data)
GC3

MUT3.data <- Read10X\_h5("3MUT\_filtered\_feature\_bc\_matrix.h5")
MUT3 <- CreateSeuratObject(counts = MUT3.data)
MUT3

# Merge more than two objects
MUT.combined <- Reduce(function(x,y) merge (x,y, all=T), list (GC1, MUT1, GC2, MUT2, GC3, MUT3))

## Part 2. Quality control:

# Add number of genes per UMI for each cell to metadata
MUT.combined$log10GenesPerUMI <- log10(MUT.combined$nFeature\_RNA) / log10(MUT.combined$nCount\_RNA)

# Compute percent mito ratio
MUT.combined$mitoRatio <- PercentageFeatureSet(object = MUT.combined, pattern = "^MT-")
MUT.combined$mitoRatio <- MUT.combined@meta.data$mitoRatio / 100

# Create .RData object
save(MUT.combined, file ="Seurat\_Object.RData")

# Filter out low quality reads using selected thresholds
filtered\_seurat <- subset(x = MUT.combined, subset= (nCount\_RNA >= 500) & (nFeature\_RNA >= 250) & (log10GenesPerUMI > 0.80) & (mitoRatio < 0.20))

# Extract counts
counts <- GetAssayData(object = filtered\_seurat, slot = "counts")

# Output a logical vector for every gene on whether the more than zero counts per cell
nonzero <- counts > 0

# Sums all TRUE values and returns TRUE if more than 10 TRUE values per gene
keep\_genes <- Matrix::rowSums(nonzero) >= 10

# Only keeping those genes expressed in more than 10 cells
filtered\_counts <- counts[keep\_genes, ]

# Reassign to filtered Seurat object
filtered\_seurat <- CreateSeuratObject(filtered\_counts, meta.data = filtered\_seurat@meta.data)

# Save.RData object
save(filtered\_seurat, file="seurat\_filtered.RData")

## Part 3. Data preparation and normalization:

#Cell cycle scoring. Normalize the counts
seurat\_phase <- NormalizeData(filtered\_seurat)

# Load cell cycle markers
load("cycle.rda")

# Score cells for cell cycle
seurat\_phase <- CellCycleScoring(seurat\_phase, g2m.features = g2m\_genes, s.features = s\_genes)

# View cell cycle scores and phases assigned to cells
View(seurat\_phase@meta.data)

# Identify the most variable genes
seurat\_phase <- FindVariableFeatures(seurat\_phase, selection.method = "vst",nfeatures = 2000, verbose = FALSE)

# Scale the counts
seurat\_phase <- ScaleData(seurat\_phase)

# Perform PCA
seurat\_phase <- RunPCA(seurat\_phase)

# Plot the PCA colored by cell cycle phase
DimPlot(seurat\_phase,reduction = "pca", group.by= "Phase", split.by = "Phase")

# Split seurat object by condition to perform cell cycle scoring and SCT on all samples
split\_seurat <- SplitObject(filtered\_seurat, split.by = "orig.ident")

for (i in 1:length(split\_seurat)) { split\_seurat[[i]] <- NormalizeData(split\_seurat[[i]], verbose = TRUE)
 split\_seurat[[i]] <- CellCycleScoring(split\_seurat[[i]], g2m.features=g2m\_genes, s.features=s\_genes)
 split\_seurat[[i]] <- SCTransform(split\_seurat[[i]], vars.to.regress = c("mitoRatio", "S.Score", "G2M.Score")) }

## Part 4. Data integration and visualization (directly after Part 3)

# Select the most variable features to use for integration
integ\_features <- SelectIntegrationFeatures(object.list = split\_seurat, nfeatures = 3000)

# Prepare the SCT list object for integration
split\_seurat <- PrepSCTIntegration(object.list = split\_seurat, anchor.features = integ\_features)

# Find anchors
integ\_anchors <- FindIntegrationAnchors(object.list = split\_seurat, normalization.method = "SCT", anchor.features = integ\_features)

# Integrate across conditions
organoids.combined.sct <- IntegrateData(anchorset = integ\_anchors, normalization.method = "SCT")

# Save integrated seurat object
saveRDS(organoids.combined.sct, "integrated\_seurat.rds")

## Run the standard workflow for visualization and clustering
DefaultAssay(organoids.combined.sct) <- "integrated"

organoids.combined.sct <- RunPCA(organoids.combined.sct, verbose = FALSE)
PCAPlot(organoids.combined.sct,split.by = "orig.ident")

organoids.combined.sct <- RunUMAP(organoids.combined.sct, reduction = "pca", dims = 1:40)

# Plot UMAP
DimPlot(organoids.combined.sct, split.by = "orig.ident")

DimPlot(organoids.combined.sct, group.by = "orig.ident")

# Elbow plot
ElbowPlot(object = organoids.combined.sct, ndims = 40)

#Find neighbors for cluster analysis
organoids.combined.sct <- FindNeighbors(organoids.combined.sct, reduction = "pca", dims = 1:40)
organoids.combined.sct <- FindClusters(organoids.combined.sct, resolution = c(0.6))

# Assign identity of clusters
Idents(object = organoids.combined.sct) <- "integrated\_snn\_res.0.6"

# Visualization
all.genes <- rownames(organoids.combined.sct)
organoids.combined.sct <- ScaleData(organoids.combined.sct, features = all.genes)

DimPlot(organoids.combined.sct, reduction = "umap", group.by = "seurat\_clusters", label = TRUE, repel = TRUE)

DimPlot(organoids.combined.sct, reduction = "umap", split.by = "orig.ident")

## Part 5. Obtain information from the datasets

#Extract identity and sample information from seurat object to determine the number of cells per cluster per sample
n\_cells <- FetchData(organoids.combined.sct, vars = c("ident", "orig.ident")) %>% dplyr::count(ident, orig.ident) %>% tidyr::spread(ident, n)

write.csv(n\_cells, "n\_cells.csv")

**## Identify conserved cell type markers**
# Switch back to the original data
DefaultAssay(organoids.combined.sct) <- "RNA"
annotations <- read.csv("annotation.txt")

# conserved markers to any cluster
get\_conserved <- function(cluster){FindConservedMarkers(organoids.combined.sct, ident.1 = cluster, grouping.var = "orig.ident", only.pos = TRUE) %>% rownames\_to\_column(var = "gene") %>% left\_join(y = unique(annotations[, c("gene\_name", "description")]), by = c("gene" = "gene\_name")) %>% cbind(cluster\_id = cluster, .)}

# Iterate function across desired clusters.
conserved\_markers <- map\_dfr(0:20, get\_conserved)

# Extract top 100 markers per cluster
top100 <- conserved\_markers %>% mutate(avg\_fc = (GC1\_avg\_log2FC + GC2\_avg\_log2FC + GC3\_avg\_log2FC + MUT1\_avg\_log2FC + MUT2\_avg\_log2FC + MUT3\_avg\_log2FC) /6) %>% group\_by(cluster\_id) %>% top\_n(n = 100, wt = avg\_fc)

write.csv(top100, "Clusters\_top100.csv")

## Identifying gene markers for each cluster

Genes <- c ("SOX2", "DCX", "TH", "NEUROD4") # Change target genes depending on the cell type

# UMAP plot
FeaturePlot(organoids.combined.sct, reduction = "umap", features = Genes, sort.cell = TRUE, min.cutoff = 'q10', max.cutoff = 5,label = T, pt.size = 0.5)

# Violin plot
plots <- VlnPlot(organoids.combined.sct, features = Genes, split.by = "orig.ident", pt.size = 0, combine = FALSE)
wrap\_plots(plots = plots, ncol = 1)

# Dot plot
DotPlot(organoids.combined.sct, features = Genes) + RotatedAxis()

## Part 6. Merge and analyse subclusters

## Combine the clusters according to the identity
new.cluster.ids <- c(1 = "Radial Glia",
 2 = "Neurons",
 3 = "NPC",
 4 = "Oligodendrocytes",
 5 = "Astrocytes", "...")

names(new.cluster.ids) <- levels(organoids.combined.sct)

organoids.combined.newnames <- RenameIdents(organoids.combined.sct, new.cluster.ids)

DimPlot(organoids.combined.newnames, reduction = "umap", label = TRUE, pt.size = 0.5) + NoLegend()

#Using DE analysis in specific clusters (after merging) MAST

annotations <- read.csv("annotation.txt")

Markers <- FindMarkers(organoids.combined.newnames, ident.1 = "GC1", ident.2 = "MUT1", group.by = "orig.ident", subset.ident = "Radial Glia", min.pct = 0.1, test.use = "MAST") %>% rownames\_to\_column(var = "gene") %>% left\_join(y = unique(annotations[, c("gene\_name", "description")]), by = c("gene" = "gene\_name"))

#Save
write.csv(Markers, "DEgenes\_cRadial Glia\_1couple.csv")

## Part 7. Create a subset of cells from a selected cluster to reanalyze

Neurons\_subset <- subset(organoids.combined.newnames, idents = "Neurons")
Neurons\_subset

DefaultAssay(Neurons\_subset) <- "integrated"

# Run the standard workflow for visualization and clustering
Neurons\_subset <- RunPCA(Neurons\_subset, verbose = FALSE)

# Plot PCA
PCAPlot(Neurons\_subset,
 split.by = "orig.ident")

#Run variable features
Neurons\_subset <- FindVariableFeatures(Neurons\_subset, selection.method = "vst", nfeatures = 2000, verbose = FALSE)

# Scale the counts
Neurons\_subset <- ScaleData(Neurons\_subset)

#Find neighbors for cluster analysis
Neurons\_subset <- FindNeighbors(Neurons\_subset, reduction = "pca", dims = 1:40)

Neurons\_subset <- FindClusters(Neurons\_subset, resolution = 0.4)

# Assign identity of clusters
Idents(object = Neurons\_subset) <- "integrated\_snn\_res.0.4"

# Visualization
all.genes <- rownames(Neurons\_subset)
Neurons\_subset <- ScaleData(Neurons\_subset, features = all.genes)

DimPlot(Neurons\_subset, reduction = "umap", group.by = "orig.ident")
DimPlot(Neurons\_subset, reduction = "umap")

# to remove a non-specific cluster
Finalcluster <- subset(Neurons\_subset, idents = 5, invert = T)
DimPlot(Finalcluster, reduction = "umap")

# To visualize the two conditions side-by-side
DimPlot(Neurons\_subset, reduction = "umap", split.by = "orig.ident")

## Cluster identification. Find conserved markers to any cluster
DefaultAssay(organoids.combined.sct) <- "RNA"

get\_conserved <- function(cluster){FindConservedMarkers(Neurons\_subset,ident.1 = cluster, grouping.var = "orig.ident",only.pos = TRUE) %>% rownames\_to\_column(var = "gene") %>% left\_join(y = unique(annotations[, c("gene\_name", "description")]), by = c("gene" = "gene\_name")) %>% cbind(cluster\_id = cluster, .)}

# Iterate function across desired clusters.
conserved\_markers <- map\_dfr(0:8, get\_conserved)

# Extract top 100 markers per cluster
top100 <- conserved\_markers %>%
 mutate(avg\_fc = (GC1\_avg\_log2FC + GC2\_avg\_log2FC + GC3\_avg\_log2FC + MUT1\_avg\_log2FC + MUT2\_avg\_log2FC + MUT3\_avg\_log2FC) /6) %>%
 group\_by(cluster\_id) %>%
 top\_n(n = 100,
 wt = avg\_fc)

#OR save
write.csv(top100, "Clusters\_top100\_Neuronsubset.csv")

## Part 8. Differential expressed genes using FIndMarkers with MAST

annotations <- read.csv("annotation.txt")

# couple 1
Markers <- FindMarkers(Neurons\_subset, ident.1 = "GC1", ident.2 = "MUT1", group.by = "orig.ident", subset.ident = "11", min.pct = 0.1, test.use = "MAST") %>%
 rownames\_to\_column(var = "gene") %>% left\_join(y = unique(annotations[, c("gene\_name", "description")]), by = c("gene" = "gene\_name"))

write.csv(Markers, "DEgenes\_C11Neuron\_1couple.csv")

# couple 2
Markers <- FindMarkers(Neurons\_subset, ident.1 = "GC2", ident.2 = "MUT2", group.by = "orig.ident", subset.ident = "11", min.pct = 0.1, test.use = "MAST") %>% rownames\_to\_column(var = "gene") %>% left\_join(y = unique(annotations[, c("gene\_name", "description")]), by = c("gene" = "gene\_name"))

write.csv(Markers, "DEgenes\_C11Neuron\_2couple.csv")

# couple 3

Markers <- FindMarkers(Neurons\_subset, ident.1 = "GC3", ident.2 = "MUT3", group.by = "orig.ident", subset.ident = "11", min.pct = 0.1, test.use = "MAST") %>%
 rownames\_to\_column(var = "gene") %>% left\_join(y = unique(annotations[, c("gene\_name", "description")]), by = c("gene" = "gene\_name"))

write.csv(Markers, "DEgenes\_C11Neuron\_3couple.csv")

## Part 9. Analysis of differential expressed genes using FIndMarkers with MAST in non-integrated samples (directly after Part 3)

# Merge datasets, example analysis for couple 1

organoids.combined.sct <- Reduce(function(x,y) merge(x,y, all=T), list (split\_seurat$GC1, split\_seurat$MUT1))
integ\_features <- SelectIntegrationFeatures(object.list = split\_seurat, nfeatures = 3000)
VariableFeatures(organoids.combined.sct[["SCT"]]) <- integ\_features

## Run the standard workflow for visualization and clustering
DefaultAssay(organoids.combined.sct) <- "integrated"

organoids.combined.sct <- RunPCA(organoids.combined.sct, verbose = FALSE)
PCAPlot(organoids.combined.sct,split.by = "orig.ident")

organoids.combined.sct <- RunUMAP(organoids.combined.sct, reduction = "pca", dims = 1:40)

# Plot UMAP
DimPlot(organoids.combined.sct, split.by = "orig.ident")

DimPlot(organoids.combined.sct, group.by = "orig.ident")

# Elbow plot
ElbowPlot(object = organoids.combined.sct, ndims = 40)

#Find neighbors for cluster analysis
organoids.combined.sct <- FindNeighbors(organoids.combined.sct, reduction = "pca", dims = 1:40)
organoids.combined.sct <- FindClusters(organoids.combined.sct, resolution = c(0.6))

# Assign identity of clusters
Idents(object = organoids.combined.sct) <- "integrated\_snn\_res.0.6"

# Visualization
all.genes <- rownames(organoids.combined.sct)
organoids.combined.sct <- ScaleData(organoids.combined.sct, features = all.genes)

DimPlot(organoids.combined.sct, reduction = "umap", group.by = "seurat\_clusters", label = TRUE, repel = TRUE)

DimPlot(organoids.combined.sct, reduction = "umap", split.by = "orig.ident")

## Continue with Part 5 and/or 6 if needed

## To analyze the clusters with resolution of 0.6
annotations <- read.csv("annotation.txt")

# Change the cluster as needed
Markers <- FindMarkers(organoids.combined.sct, ident.1 = "GC1", ident.2 = "MUT1", group.by = "orig.ident", subset.ident = "20", min.pct = 0.1, test.use = "MAST") %>% rownames\_to\_column(var = "gene") %>% left\_join(y = unique(annotations[, c("gene\_name", "description")]), by = c("gene" = "gene\_name"))

#save
write.csv(Markers, "DEgenes\_Cluster20\_couple1.csv")