R studio script for PCA analysis of NGS data from Tet-On LLCs (Fig. 1g)

install.packages("plotrix")

library("ggplot2")

library("plotly")

library("plotrix")

library("ggrepel")

data <- read.csv("231005\_Sato\_et\_al\_processed\_data.csv", header = T, row.names = 1)

data <- data[,c(1,2,3,4,5,6)]

head(data)

data <- t(data)

data <- data[, apply(data, 2, var) !=0]

data.pca <- prcomp(data, scale = T)

summary(data.pca)

data.pca$x

explained\_variance <- summary(data.pca)$importance[2, ]

explained\_variance\_percent <- round(explained\_variance\*100, 2)

data.g <- as.data.frame(data.pca$x)

group <- c("iPSC", "TetON\_day6", "TetON\_day14", "TetON\_day20", "TetON\_day30", "TetON\_day40")

g <- ggplot(data.g, aes(x = PC1, y = PC2, label = rownames(data.g), color = group)) +

labs(x = paste("PC1 (", explained\_variance\_percent[1], "%)", sep = ""),

y = paste("PC2 (", explained\_variance\_percent[2], "%)", sep = "")) +

geom\_point(size = 3) +

geom\_text\_repel(color = "black", size = 4) +

theme\_linedraw() +

theme(legend.position = "none") +

scale\_x\_continuous(limits = c(-200, 100)) +

scale\_y\_continuous(limits = c(-200, 100)) +

scale\_color\_manual(values = c("iPSC" = "#c4ebcb",

"TetON\_day6" = "#eeeab7",

"TetON\_day14" = "#fbdc9c",

"TetON\_day20" = "#f89f94",

"TetON\_day30" = "#fa80b5",

"TetON\_day40" = "#d3add9"))

g

R studio script for PCA analysis of NGS data from Tet-Off LLCs (Supplementary Fig. 4f)

install.packages("plotrix")

library("ggplot2")

library("plotly")

library("plotrix")

library("ggrepel")

data <- read.csv("231005\_Sato\_et\_al\_processed\_data.csv", header = T, row.names = 1)

data <- data[,c(1,2,5,7)]

head(data)

data <- t(data)

data <- data[, apply(data, 2, var) !=0]

data.pca <- prcomp(data, scale = T)

summary(data.pca)

data.pca$x

explained\_variance <- summary(data.pca)$importance[2, ]

explained\_variance\_percent <- round(explained\_variance\*100, 2)

data.g <- as.data.frame(data.pca$x)

group <- c("iPSC", "TetON\_day6", "TetON\_day30","TetOFF\_day35")

g <- ggplot(data.g, aes(x = PC1, y = PC2, label = rownames(data.g), color = group)) +

labs(x = paste("PC1 (", explained\_variance\_percent[1], "%)", sep = ""),

y = paste("PC2 (", explained\_variance\_percent[2], "%)", sep = "")) +

geom\_point(size = 3) +

geom\_text\_repel(color = "black", size = 4) +

theme\_linedraw() +

theme(legend.position = "none") +

scale\_x\_continuous(limits = c(-150, 100)) +

scale\_y\_continuous(limits = c(-150, 100)) +

scale\_color\_manual(values = c("iPSC" = "#c4ebcb",

"TetON\_day6" = "#fcd9ac",

"TetON\_day30" = "#e87c11",

"TetOFF\_day35" ="#44acf4"))

g

R studio script for scRNA-seq analysis (Fig. 3a-f, Supplementary Fig. 5a-c)

library(Seurat)

library(dplyr)

library(Matrix)

library(gtable)

library(grid)

library(gridExtra)

library(rlang)

library(patchwork)

library(tidyr)

library(pals)

data.t <- read.delim("GSE112013\_Combined\_UMI\_table.txt", header = T, row.names = 1)

data.s <- read.delim("matrix\_inflection\_demulti\_DBEC\_KoyanagiWTA2.txt.gz", header = T, row.names = 1)

for (i in c("hST02", "hST03", "hST04", "hST05", "hST06")) {

assign(sprintf("data.%s", i), select(data.s, starts\_with(i)))}

data.s <- cbind(data.hST03, data.hST06)

Adult <- CreateSeuratObject(counts = data.t, project = "testis", min.cells =3, min.features = 500)

data.S <- CreateSeuratObject(counts = data.s, project = "diff", min.cells =3, min.features = 500)

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

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

VlnPlot(Adult, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mt"), ncol = 3)

VlnPlot(data.S, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mt"), ncol = 3)

Adult <- subset(Adult, subset = nFeature\_RNA > 500 & nFeature\_RNA < 5000 & percent.mt < 10)

data.S <- subset(data.S, subset = nFeature\_RNA > 5000 & nFeature\_RNA < 10000 & percent.mt < 20)

VlnPlot(Adult, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mt"), ncol = 3)

VlnPlot(data.S, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mt"), ncol = 3)

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

data.S <- NormalizeData(data.S, normalization.method = "LogNormalize", scale.factor = 10000)

Adult <- FindVariableFeatures(Adult, selection.method = "vst", nfeatures = 2000)

data.S <- FindVariableFeatures(data.S, selection.method = "vst", nfeatures = 2000)

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

top10 <- head(VariableFeatures(data.S), 10)

plot1 <- VariableFeaturePlot(Adult)

plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)

plot1 + plot2

plot1s <- VariableFeaturePlot(data.S)

plot2s <- LabelPoints(plot = plot1s, points = top10, repel = TRUE)

plot1s + plot2s

all.genes <- rownames(Adult)

all.genes.s <- rownames(data.S)

Adult <- ScaleData(Adult, features = all.genes)

data.S <- ScaleData(data.S, features = all.genes.s)

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

print(Adult[["pca"]], dims = 1:5, nfeatures = 5)

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

DimPlot(Adult, reduction = "pca")

DimHeatmap(Adult, dims = 2, cells = 500, balanced = TRUE)

DimHeatmap(Adult, dims = 1:15, cells = 500, balanced = TRUE)

data.S <- RunPCA(data.S, features = VariableFeatures(object = data.S))

print(data.S[["pca"]], dims = 1:5, nfeatures = 5)

VizDimLoadings(data.S, dims = 1:2, reduction = "pca")

DimPlot(data.S, reduction = "pca")

DimHeatmap(data.S, dims = 2, cells = 500, balanced = TRUE)

DimHeatmap(data.S, dims = 1:15, cells = 500, balanced = TRUE)

ElbowPlot(Adult)

ElbowPlot(data.S)

Adult <- FindNeighbors(Adult, dims = 1:10)

Adult <- FindClusters(Adult, resolution = 0.15)

data.S <- FindNeighbors(data.S, dims = 1:10)

data.S <- FindClusters(data.S, resolution = 0.15)

head(Idents(Adult), 5)

head(Idents(data.S), 5)

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

p1<-DimPlot(Adult, reduction = "umap")

p2<-DimPlot(Adult, reduction = "umap")

p1+p2

FeaturePlot(Adult, features = c("DLK1","MYH11","VWF","CD163","SOX9","DDX4"))

data.S <- RunUMAP(data.S, dims = 1:10)

p1s<-DimPlot(data.S, reduction = "umap")

p2s<-DimPlot(data.S, reduction = "umap")

p1s+p2s

FeaturePlot(data.S, features = c("DLK1","MYH11","VWF","CD163","MYH11","SOX9","DDX4"))

data <- merge(data.S, y=c(Adult),

add.cell.ids = c("data.S","Adult"),

project = "testis",

merge.data=T)

data.list <- SplitObject(data, split.by = "orig.ident")

data.list <- lapply(X = data.list, FUN = function(x) {

x <- NormalizeData(x)

x <- FindVariableFeatures(x, selection.method = "vst", nfeatures = 10000)})

features <- SelectIntegrationFeatures(object.list = data.list, nfeatures = 10000)

data.anchors <- FindIntegrationAnchors(object.list = data.list, anchor.features = features)

data.combined <- IntegrateData(anchorset = data.anchors)

DefaultAssay(data.combined) <- "integrated"

data.combined <- ScaleData(data.combined, verbose = F)

data.combined <- RunPCA(data.combined, npcs = 50, verbose = F)

data.combined <- RunUMAP(data.combined, reduction = "pca", dims = 1:50)

data.combined <- FindNeighbors(data.combined, reduction = "pca", dims = 1:50)

data.combined <- FindClusters(data.combined, resolution = 0.15)

# [Fig3B]

p1 <- DimPlot(data.combined, reduction = "umap", group.by = "orig.ident")

p2 <- DimPlot(data.combined, reduction = "umap", label = T, repel = T)

p1+p2

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

# [S Fig5A]

FeaturePlot(data.combined, features = c("DLK1"),min.cutoff = "q9")

FeaturePlot(data.combined, features = c("MYH11"),min.cutoff = "q9")

FeaturePlot(data.combined, features = c("CD163"),min.cutoff = "q9")

FeaturePlot(data.combined, features = c("SOX9"),min.cutoff = "q9")

FeaturePlot(data.combined, features = c("DDX4"),min.cutoff = "q9")

FeaturePlot(data.combined, features = c("VWF"),min.cutoff = "q9")

# [Fig3C]

DotPlot(data.combined, features = c("DLK1","LHCGR","STAR","CYP17A1","CYP11A1","INSL3","HSD3B1","IGF1","HSD17B3"))

# [Fig3D]

VlnPlot(data.combined, features = c("DLK1"))

VlnPlot(data.combined, features = c("IGF1"))

VlnPlot(data.combined, features = c("IGF2"))

VlnPlot(data.combined, features = c("STAR"))

# find markers for every cluster compared to all remaining cells, report only the positive ones

data.combined.markers <- FindAllMarkers(data.combined, only.pos = TRUE, min.pct = 0.25, logfc.threshold = 0.25)

data.combined.markers %>%

group\_by(cluster) %>%

slice\_max(n = 2, order\_by = avg\_log2FC)

# heatmap [Fig3E\_left]

data.combined.markers %>%

group\_by(cluster) %>%

top\_n(n = 100, wt = avg\_log2FC) -> top100

DoHeatmap(data.combined, features = top100$gene)

# [Fig3E\_right]

library(dplyr)

data.combined.markers %>%

group\_by(cluster) %>%

top\_n(n = 100, wt = avg\_log2FC) %>%

select(cluster, gene) %>%

group\_walk(~ print(.x))

# cluster 0(LLC subset 1) > cluster 9(LLC susbset 2) [S Fig 5B]

cluster0.d.from.9.markers <- FindMarkers(data.combined, ident.1 = 0, ident.2 = c(9), min.pct = 0.25)

cluster0.upregulated <- cluster0.d.from.9.markers[cluster0.d.from.9.markers$avg\_log2FC > 0, ]

head(cluster0.upregulated, n = 100)

# cluster 9(LLC subset 2) > cluster 0(LLC susbset 1) [S Fig 5B]

cluster9.d.from.0.markers <- FindMarkers(data.combined, ident.1 = 9, ident.2 = c(0), min.pct = 0.25)

cluster9.upregulated <- cluster9.d.from.0.markers[cluster9.d.from.0.markers$avg\_log2FC > 0, ]

head(cluster9.upregulated, n = 100)

# [S Fig 5C]

DotPlot(data.combined, features = c("MKI67", "PCNA","CCND1","CCNE1","CCNA2","CCNB1","CDC20","CDK1","MCM2"))

# [Fig 5F]

DotPlot(data.combined, features = c("CYP11A1", "PDGFRA","IGF2","NES","PDGFRB","TMSB10"))

R studio script for scRNA-seq analysis (Supplementary Fig. 5d, e)

library(Seurat)

library(dplyr)

library(Matrix)

library(gtable)

library(grid)

library(gridExtra)

library(rlang)

library(patchwork)

library(tidyr)

library(pals)

data.s <- read.delim("matrix\_inflection\_demulti\_DBEC\_KoyanagiWTA2.txt.gz",

header = T, row.names = 1)

for (i in c("hST01", "hST02", "hST03", "hST04")) {

assign(sprintf("data.%s", i), select(data.s, starts\_with(i)))}

data.s <- cbind(data.hST01, data.hST02, data.hST03, data.hST04)

diff <- CreateSeuratObject(counts = data.s, project = "Leydig\_dif", min.cells =3, min.features = 500)

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

VlnPlot(diff, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mt"), ncol = 3)

diff <- subset(diff, subset = nFeature\_RNA > 5000 & nFeature\_RNA < 10000 & percent.mt < 13)

VlnPlot(diff, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mt"), ncol = 3)

diff@meta.data$time <-"diff"

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

merge.data <- FindVariableFeatures(diff, selection.method = "vst", nfeatures = 2000)

top10 <- head(VariableFeatures(merge.data), 10)

plot1 <- VariableFeaturePlot(merge.data)

plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)

plot1 + plot2

all.genes <- rownames(merge.data)

merge.data <- ScaleData(merge.data, features = all.genes)

merge.data <- RunPCA(merge.data, features = VariableFeatures(object = merge.data))

print(merge.data[["pca"]], dims = 1:5, nfeatures = 5)

VizDimLoadings(merge.data, dims = 1:2, reduction = "pca")

DimPlot(merge.data, reduction = "pca")

DimHeatmap(merge.data, dims = 2, cells = 500, balanced = TRUE)

DimHeatmap(merge.data, dims = 1:15, cells = 500, balanced = TRUE)

ElbowPlot(merge.data)

merge.data <- FindNeighbors(merge.data, dims = 1:15)

merge.data <- FindClusters(merge.data, resolution = 0.4)

head(Idents(merge.data), 5)

merge.data <- RunUMAP(merge.data, dims = 1:15)

sample\_colors <- c("hST01" = "#b6d2bb", "hST02" = "#69cbd8", "hST03" = "#e8a0c1", "hST04" = "#fac18e")

# [S Fig5D]

p1 <- DimPlot(merge.data, reduction = "umap", group.by = "time", cols = sample\_colors[merge.data$orig.ident])

p2 <- DimPlot(merge.data, reduction = "umap", group.by = "orig.ident", cols = sample\_colors[merge.data$orig.ident])

p2

#[S Fig5E]

FeaturePlot(merge.data, features = c("DLK1"))

FeaturePlot(merge.data, features = c("SOX17"))

FeaturePlot(merge.data, features = c("ACTA2"))

FeaturePlot(merge.data, features = c("SOX1"))