Single-nuclei RNA-seq and matched CAG sizing - ELongATE

Index

Load_packages

In [ ]:
```
library("dplyr")
library("Seurat")
library("patchwork")
library("Matrix")
library("biomaRt")
library("ggplot2")
library("celda")
library("DoubletFinder")
library("hdf5r")
library("harmony")
library("presto")
library("neuralnet")
library("keras")
library("tensorflow")
library("glmnet")
library("caret")
library("descriptio")
library("gridExtra")
library("ggpubr")
library("weights")
setwd("/path/to/results/")
```

Read_Handsaker_metadata

In [ ]:
```
pb_metadata_file <- "PROJECT_ELongATE/Handsaker/Analysis_bag_6_Broad_Huntingtons_Caudate_2024_PacBio_Metadata_Open/pacbio_deepdives_cells_unfiltered.txt"
pb_metadata <- read.table(pb_metadata_file, sep = "\t", header = TRUE)
sample_name <- gsub(pattern = '(.*)_\\w+', replacement = '\\1', pb_metadata$CELL_BARCODE)
CB <- gsub(pattern = '.*_(\\w+)', replacement = '\\1', pb_metadata$CELL_BARCODE)
pb_metadata <- cbind(pb_metadata, CB, sample_name)
rownames(pb_metadata) <- gsub(x = pb_metadata$CELL_BARCODE, pattern = "Caudate", replacement = "HD_Caudate")
```

In [ ]:
```
tmp <- lapply(split(pb_metadata, pb_metadata$DONOR), function(x) {
    if (x$DONOR[1] != "S04577") {
      x <- x[which(x$CAGLENGTH > 36), ]
    }
    num_cells <- dim(x)[1]
    num_spns <- dim(x[which(x$CELLTYPE == "SPN"), ])[1]
    return(data.frame(num_cells, num_spns))
})

num_cells_Handsaker <- do.call(rbind, tmp)
#num_cells_Handsaker
```

Read_Handsaker_expression_matrix

In [ ]:
```
h5_files <- list.files(path = "PROJECT_ELongATE/Handsaker/", pattern = "\\.
h5$", full.names = TRUE)
names(h5_files) <- gsub(x = basename(h5_files), pattern = "\\.umi.*", repla
cement = "")
```

In [ ]:
```
Read_h5_file = function(h5_file, metadata) {
    #extract counts
    counts <- Read10X_h5(filename = h5_file, use.names = TRUE)
    #create a single cell experiment object
    sce <- SingleCellExperiment(list(counts = counts))
    #create a seurat object
    seuratObject <- CreateSeuratObject(counts(sce), project = names(h5_fil
e), min.cells = 0, min.features = 0)
    seuratObject[["CB"]] <- rownames(seuratObject@meta.data)
    #print(head(rownames(metadata)))
    #print(head(seuratObject@meta.data))
    #create tmp variable and edit rownames
    tmp <- seuratObject@meta.data
    rownames(tmp) <- gsub(x = gsub(x = paste0(seuratObject@meta.data$orig.i
dent, "_",
                                              gsub(x = seuratObject@meta.da
ta$CB, pattern = "_.*", replacement = "")),
                                    pattern = "_10X_", replacement = "_"),
                          pattern = "_merged_", replacement = "_")
    #exploit edited rownames to add metadata to Seurat object
    tmp <- cbind(tmp, metadata[rownames(tmp), ])
    #print(head(rownames(tmp)))
    #replace row names with old ones
    rownames(tmp) <- rownames(seuratObject@meta.data)
    #update original object
    seuratObject@meta.data <- tmp
    #print(head(seuratObject@meta.data))
    #print(head(rownames(seuratObject@meta.data)))
    #retain only SPN
    seuratObject <- subset(x = seuratObject, subset = CELLTYPE == "SPN")
    #run SCTransform on SPN only
    seuratObject <- SCTransform(seuratObject)
    cat(sprintf("%s; Num. features = %d; Num. SPN = %d\n", names(h5_file),
dim(seuratObject)[1], dim(seuratObject)[2]))
    return(seuratObject)
}
```

In [ ]:
```
names(h5_files)
```

In [ ]:
```
# S02205_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S02205_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S02205_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S02205_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S02205_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S02205_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)
# S02205_HD_Caudate_DeepDive_rxn4 <- Read_h5_file(h5_files["S02205_HD_Cauda
te_DeepDive_rxn4"], pb_metadata)

# S02205 <- merge(x = S02205_HD_Caudate_DeepDive_rxn1,
#                 y = c(S02205_HD_Caudate_DeepDive_rxn2,
#                       S02205_HD_Caudate_DeepDive_rxn3,
#                       S02205_HD_Caudate_DeepDive_rxn4))

# saveRDS(object = S02205, file = "S02205.rds")
```

In [ ]:
```
# S04002_10X_rxn1 <- Read_h5_file(h5_files["S04002_10X_rxn1"], pb_metadata)
# S04002_10X_rxn2_merged <- Read_h5_file(h5_files["S04002_10X_rxn2_merge
d"], pb_metadata)
# S04002_10X_rxn3_merged <- Read_h5_file(h5_files["S04002_10X_rxn3_merge
d"], pb_metadata)
# S04002_10X_rxn4_merged <- Read_h5_file(h5_files["S04002_10X_rxn4_merge
d"], pb_metadata)
# S04002_rxn5 <- Read_h5_file(h5_files["S04002_rxn5"], pb_metadata)
# S04002_rxn6 <- Read_h5_file(h5_files["S04002_rxn6"], pb_metadata)
# S04002_rxn7 <- Read_h5_file(h5_files["S04002_rxn7"], pb_metadata)
# S04002_rxn8 <- Read_h5_file(h5_files["S04002_rxn8"], pb_metadata)

# S04002 <- merge(x = S04002_10X_rxn1,
#                 y = c(S04002_10X_rxn2_merged,
#                       S04002_10X_rxn3_merged,
#                       S04002_10X_rxn4_merged,
#                       S04002_rxn5,
#                       S04002_rxn6,
#                       S04002_rxn7,
#                       S04002_rxn8))

# saveRDS(object = S04002, file = "S04002.rds")
```

In [ ]:
```
# S04577_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S04577_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S04577_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S04577_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S04577_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S04577_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)
# S04577_HD_Caudate_DeepDive_rxn4 <- Read_h5_file(h5_files["S04577_HD_Cauda
te_DeepDive_rxn4"], pb_metadata)

# S04577 <- merge(x = S04577_HD_Caudate_DeepDive_rxn1,
#                 y = c(S04577_HD_Caudate_DeepDive_rxn2,
#                       S04577_HD_Caudate_DeepDive_rxn3,
#                       S04577_HD_Caudate_DeepDive_rxn4))

# saveRDS(object = S04577, file = "S04577.rds")
```

In [ ]:
```
# S05202_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S05202_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S05202_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S05202_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S05202_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S05202_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)

# S05202 <- merge(x = S05202_HD_Caudate_DeepDive_rxn1,
#                 y = c(S05202_HD_Caudate_DeepDive_rxn2,
#                       S05202_HD_Caudate_DeepDive_rxn3))

# saveRDS(object = S05202, file = "S05202.rds")
```

In [ ]:
```
# S05368_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S05368_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S05368_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S05368_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S05368_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S05368_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)

# S05368 <- merge(x = S05368_HD_Caudate_DeepDive_rxn1,
#                 y = c(S05368_HD_Caudate_DeepDive_rxn2,
#                       S05368_HD_Caudate_DeepDive_rxn3))

# saveRDS(object = S05368, file = "S05368.rds")
```

In [ ]:
```
# S06758_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S06758_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S06758_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S06758_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S06758_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S06758_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)
# S06758_HD_Caudate_DeepDive_rxn4 <- Read_h5_file(h5_files["S06758_HD_Cauda
te_DeepDive_rxn4"], pb_metadata)
# S06758_HD_Caudate_DeepDive_rxn5 <- Read_h5_file(h5_files["S06758_HD_Cauda
te_DeepDive_rxn5"], pb_metadata)

# S06758 <- merge(x = S06758_HD_Caudate_DeepDive_rxn1,
#                 y = c(S06758_HD_Caudate_DeepDive_rxn2,
#                       S06758_HD_Caudate_DeepDive_rxn3,
#                       S06758_HD_Caudate_DeepDive_rxn4,
#                       S06758_HD_Caudate_DeepDive_rxn5))

# saveRDS(object = S06758, file = "S06758.rds")
```

In [ ]:
```
# S07681_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S07681_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S07681_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S07681_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S07681_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S07681_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)
# S07681_HD_Caudate_DeepDive_rxn4 <- Read_h5_file(h5_files["S07681_HD_Cauda
te_DeepDive_rxn4"], pb_metadata)
# S07681_HD_Caudate_DeepDive_rxn5 <- Read_h5_file(h5_files["S07681_HD_Cauda
te_DeepDive_rxn5"], pb_metadata)
# S07681_HD_Caudate_DeepDive_rxn6 <- Read_h5_file(h5_files["S07681_HD_Cauda
te_DeepDive_rxn6"], pb_metadata)

# S07681 <- merge(x = S07681_HD_Caudate_DeepDive_rxn1,
#                 y = c(S07681_HD_Caudate_DeepDive_rxn2,
#                       S07681_HD_Caudate_DeepDive_rxn3,
#                       S07681_HD_Caudate_DeepDive_rxn4,
#                       S07681_HD_Caudate_DeepDive_rxn5,
#                       S07681_HD_Caudate_DeepDive_rxn6))

# saveRDS(object = S07681, file = "S07681.rds")
```

In [ ]:
```
# S09619_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S09619_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S09619_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S09619_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S09619_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S09619_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)
# S09619_HD_Caudate_DeepDive_rxn4 <- Read_h5_file(h5_files["S09619_HD_Cauda
te_DeepDive_rxn4"], pb_metadata)

# S09619 <- merge(x = S09619_HD_Caudate_DeepDive_rxn1,
#                 y = c(S09619_HD_Caudate_DeepDive_rxn2,
#                       S09619_HD_Caudate_DeepDive_rxn3,
#                       S09619_HD_Caudate_DeepDive_rxn4))

# saveRDS(object = S09619, file = "S09619.rds")
```

In [ ]:
```
# S12365_HD_Caudate_DeepDive_rxn1 <- Read_h5_file(h5_files["S12365_HD_Cauda
te_DeepDive_rxn1"], pb_metadata)
# S12365_HD_Caudate_DeepDive_rxn2 <- Read_h5_file(h5_files["S12365_HD_Cauda
te_DeepDive_rxn2"], pb_metadata)
# S12365_HD_Caudate_DeepDive_rxn3 <- Read_h5_file(h5_files["S12365_HD_Cauda
te_DeepDive_rxn3"], pb_metadata)
# S12365_HD_Caudate_DeepDive_rxn4 <- Read_h5_file(h5_files["S12365_HD_Cauda
te_DeepDive_rxn4"], pb_metadata)

# S12365 <- merge(x = S12365_HD_Caudate_DeepDive_rxn1,
#                 y = c(S12365_HD_Caudate_DeepDive_rxn2,
#                       S12365_HD_Caudate_DeepDive_rxn3,
#                       S12365_HD_Caudate_DeepDive_rxn4))

# saveRDS(object = S12365, file = "S12365.rds")
```

In [ ]:

```
# SPN_all <- merge(x = S02205_HD_Caudate_DeepDive_rxn1,
#                  y = c(S02205_HD_Caudate_DeepDive_rxn2,
#                        S02205_HD_Caudate_DeepDive_rxn3,
#                        S02205_HD_Caudate_DeepDive_rxn4,
#                        S04002_10X_rxn1,
#                        S04002_10X_rxn2_merged,
#                        S04002_10X_rxn3_merged,
#                        S04002_10X_rxn4_merged,
#                        S04002_rxn5,
#                        S04002_rxn6,
#                        S04002_rxn7,
#                        S04002_rxn8,
#                        S04577_HD_Caudate_DeepDive_rxn1,
#                        S04577_HD_Caudate_DeepDive_rxn2,
#                        S04577_HD_Caudate_DeepDive_rxn3,
#                        S04577_HD_Caudate_DeepDive_rxn4,
#                        S05202_HD_Caudate_DeepDive_rxn1,
#                        S05202_HD_Caudate_DeepDive_rxn2,
#                        S05202_HD_Caudate_DeepDive_rxn3,
#                        S05368_HD_Caudate_DeepDive_rxn1,
#                        S05368_HD_Caudate_DeepDive_rxn2,
#                        S05368_HD_Caudate_DeepDive_rxn3,
#                        S06758_HD_Caudate_DeepDive_rxn1,
#                        S06758_HD_Caudate_DeepDive_rxn2,
#                        S06758_HD_Caudate_DeepDive_rxn3,
#                        S06758_HD_Caudate_DeepDive_rxn4,
#                        S06758_HD_Caudate_DeepDive_rxn5,
#                        S07681_HD_Caudate_DeepDive_rxn1,
#                        S07681_HD_Caudate_DeepDive_rxn2,
#                        S07681_HD_Caudate_DeepDive_rxn3,
#                        S07681_HD_Caudate_DeepDive_rxn4,
#                        S07681_HD_Caudate_DeepDive_rxn5,
#                        S07681_HD_Caudate_DeepDive_rxn6,
#                        S09619_HD_Caudate_DeepDive_rxn1,
#                        S09619_HD_Caudate_DeepDive_rxn2,
#                        S09619_HD_Caudate_DeepDive_rxn3,
#                        S09619_HD_Caudate_DeepDive_rxn4,
#                        S12365_HD_Caudate_DeepDive_rxn1,
#                        S12365_HD_Caudate_DeepDive_rxn2,
#                        S12365_HD_Caudate_DeepDive_rxn3,
#                        S12365_HD_Caudate_DeepDive_rxn4))

# #Add metadata
# SPN_all[["percent.mt"]] <- PercentageFeatureSet(SPN_all, pattern = "^MT-
")
# SPN_all[["SAMPLE"]] <- factor(unlist(lapply(strsplit(SPN_all@meta.data$sa
mple_name, "_"), '[[', 1)))
# SPN_all[["EXP_CAGLENGTH"]] <- NA
# SPN_all[["PHASE"]] <- NA
# SPN_all@meta.data$EXP_CAGLENGTH[which(SPN_all@meta.data$CAGLENGTH > 37)]
<- SPN_all@meta.data$CAGLENGTH[which(SPN_all@meta.data$CAGLENGTH > 37)]
# SPN_all@meta.data$PHASE[which(SPN_all@meta.data$EXP_CAGLENGTH < 150)] <-
"A-B"
# SPN_all@meta.data$PHASE[which(SPN_all@meta.data$EXP_CAGLENGTH >= 150)] <-
"C-D-E"

# head(SPN_all@meta.data)

# # SPN_all <- merge(x = S02205,
# #                  y = c(S04002,
```

```
# #                         S04577,
# #                         S05202,
# #                         S05368,
# #                         S06758,
# #                         S07681,
# #                         S09619,
# #                         S12365),
# #                   add.cell.ids = c("S02205", "S04002", "S04577", "S0520
2", "S05368", "S06758", "S07681", "S09619", "S12365"),
# #                    project = "snRNAseqCAG")

# saveRDS(object = SPN_all, file = "SPN_all.rds")
```

In [ ]:
```
#Load dataset with SPNs from all samples
SPN_all <- readRDS("SPN_all.rds")
```

In [ ]:
```
str(SPN_all@meta.data)
```

In [ ]:
```
sort(table(SPN_all@meta.data$orig.ident))
#sort(table(SPN_all@meta.data$orig.ident[grep(x = SPN_all@meta.data$orig.id
ent, pattern = "S05202")]))
```

In [ ]:
```
#find total number of SPNs and ratio of SPNs with Wt-allele only
tot <- table(unlist(lapply(strsplit(SPN_all@meta.data$sample_name, "_"),
'[[', 1)))
num_WT <- table(unlist(lapply(strsplit(SPN_all@meta.data[which(SPN_all@met
a.data$CAGLENGTH < 36), "sample_name"], "_"), '[[', 1)))
num_HD <- table(unlist(lapply(strsplit(SPN_all@meta.data[which(SPN_all@met
a.data$CAGLENGTH > 36), "sample_name"], "_"), '[[', 1)))
num_HD_exp <- table(unlist(lapply(strsplit(SPN_all@meta.data[which(SPN_all@
meta.data$CAGLENGTH > 150), "sample_name"], "_"), '[[', 1)))

tot
num_WT/tot[names(num_WT)]
num_HD/tot[names(num_HD)]
num_HD_exp/tot[names(num_HD_exp)]
num_HD_exp/num_HD[names(num_HD_exp)]
```

In [ ]:
```
# head(S02205@meta.data)
# head(S04002@meta.data)
# head(S04577@meta.data)
# head(S05202@meta.data)
# head(S05368@meta.data)
# head(S06758@meta.data)
# head(S07681@meta.data)
# head(S09619@meta.data)
# head(S12365@meta.data)
head(SPN_all@meta.data)
SPN_all
```

Filter_Handsaker_SPN_UMAP

In [ ]:
```
#Run SCTransform
SPN_all <- SCTransform(SPN_all, variable.features.n = 3000,  ncells = 5000)
#Run PCA
SPN_all <- RunPCA(SPN_all, features = VariableFeatures(object = SPN_all), n
pcs = 50, reduction.name = "pca")
ElbowPlot(SPN_all, reduction = "pca", ndims = 50)
num_dims <- 40
#Run UMAP
SPN_all <- RunUMAP(SPN_all, dims = 1:num_dims, reduction = "pca", reductio
n.name = "umap", reduction.key = "umap")
```

In [ ]:
```
#do plots
options(repr.plot.width=20, repr.plot.height=20)
FeaturePlot(SPN_all, reduction = "umap", features = "CAGLENGTH", cols = c
("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_CAGLENGTH.pdf", width = 8, height = 8)
FeaturePlot(SPN_all, reduction = "umap", features = "EXP_CAGLENGTH", cols =
c("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_EXP_CAGLENGTH.pdf", width = 8, height = 8)
DimPlot(SPN_all, reduction = "umap", group.by = "PHASE", pt.size = 1)
ggsave("UMAP_Handsaker_PHASE.pdf", width = 8, height = 8)
FeaturePlot(SPN_all, reduction = "umap", features = "nCount_RNA", cols = c
("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_NCOUNTS.pdf", width = 8, height = 8)
FeaturePlot(SPN_all, reduction = "umap", features = "nFeature_RNA", cols =
c("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_NFEATURES.pdf", width = 8, height = 8)
FeaturePlot(SPN_all, reduction = "umap", features = "percent.mt", cols = c
("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_PERCMT.pdf")
DimPlot(SPN_all, group.by = "SAMPLE", reduction = "umap", pt.size = 1)
ggsave("UMAP_Handsaker_splitBySampleMerged.pdf", width = 8, height = 8)
DimPlot(SPN_all, split.by = "SAMPLE", reduction = "umap", pt.size = 1, ncol
= 3, group.by = "SAMPLE") + NoLegend()
ggsave("UMAP_Handsaker_splitBySample.pdf", width = 20, height = 20)
```

In [ ]:
```
#do clustering
SPN_all <- FindNeighbors(SPN_all, reduction = "pca", dims = 1:num_dims)
SPN_all <- FindClusters(SPN_all, resolution = 0.01, method = 4)
DimPlot(SPN_all, reduction = "umap", group.by = "seurat_clusters", pt.size
= 1)
ggsave("UMAP_Handsaker_clusters.pdf", width = 8, height = 8)
```

In [ ]:
```r
#find gene markers for the two clusters
SPN_all <- PrepSCTFindMarkers(SPN_all, assay = "SCT", verbose = TRUE)

# find markers for every cluster compared to all remaining cells, report on
ly the positive ones
SPN_all.markers <- FindAllMarkers(SPN_all, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25, assay = "SCT")

#find the top marker for each cluster and plot their expression values
SPN_all_top_marker <- SPN_all.markers %>%
  group_by(cluster) %>%
  slice_max(n = 1, order_by = avg_log2FC)

# num_genes_plot <- 10
# if (length(SPN_all_top_marker) < num_genes_plot) {
#   chunks_genes <- list(1:length(SPN_all_top_marker))
# } else {
#   chunks_genes <- split(1:length(SPN_all_top_marker), ceiling(seq(from =
1, to = length(SPN_all_top_marker))/num_genes_plot))
# }

# #plot normalized counts
# for (i in 1:length(chunks_genes)) {
#     p <- VlnPlot(SPN_all, features = SPN_all_top_marker$gene[chunks_genes
[[i]]])
#     #+ ggtitle("Normalized counts for top markers")
#     plot(p)
#     ggsave(paste0("Handsaker_Normalized_MarkerGeneCounts_chunk", i, ".pd
f"), device = pdf, height = 8, width = 8)
# }
```

In [ ]:
```r
#extract top 10 markers for each cluster and plot heatmap
SPN_all_top_10_markers <- SPN_all.markers %>%
  group_by(cluster) %>%
  top_n(n = 10, wt = avg_log2FC)

p <- DoHeatmap(SPN_all, features = SPN_all_top_10_markers$gene) + NoLegend
() + ggtitle("Heatmap with top 10 marker genes per cluster")
plot(p)
ggsave("Handsaker_Heatmap_top10MarkerGeneCounts.pdf", device = pdf, height
= 8, width = 16)
```

In [ ]:
```r
SPN_all_top_10_markers
```

In [ ]:
```r
#read phase C-D file with all genes
phaseCD_all_file <- "PROJECT_ELongATE/PhaseCD_all.txt"

phaseCD_genes_all <- read.table(phaseCD_all_file, sep = "\t", header = TRU
E)
rownames(phaseCD_genes_all) <- phaseCD_genes_all$Gene

phaseCD_genes_all[intersect(rownames(phaseCD_genes_all), SPN_all_top_10_mar
kers[which(SPN_all_top_10_markers$cluster == 1), ]$gene), ]
```

In [ ]:
```r
#create file with only phase C-, C+ and D genes
phaseCD_file <- "PROJECT_ELongATE/PhaseCD.txt"
system(command = paste0("cat ", phaseCD_all_file, " | awk 'BEGIN {FS=\"\t
\"} { if ($8 != NA) print }' > ", phaseCD_file))
```

In [ ]:
```
#filter only SPNs from cluster 0
SPN_all_unfiltered <- SPN_all
SPN_all <- subset(x = SPN_all, subset = seurat_clusters == "0")
```

In [ ]:
```
# saveRDS(object = SPN_all, file = "SPN_filtered.rds")
```

In [ ]:
```
# SPN_all <- readRDS("SPN_filtered.rds")
```

In [ ]:
```
#sample S04577 is CTRL -> filter out from HD samples
SPN_HD <- subset(x = SPN_all, subset = SAMPLE != "S04577")
#Run PCA and UMAP on HD samples
SPN_HD <- RunPCA(SPN_HD, features = VariableFeatures(object = SPN_HD), npcs
= 50, reduction.name = "pca")
SPN_HD <- RunUMAP(SPN_HD, dims = 1:num_dims, reduction = "pca", reduction.n
ame = "umap", reduction.key = "umap")
#Run PCA and UMAP on CTRL sample
SPN_CTRL <- subset(x = SPN_all, subset = SAMPLE == "S04577")
SPN_CTRL <- RunPCA(SPN_CTRL, features = VariableFeatures(object = SPN_CTR
L), npcs = 50, reduction.name = "pca")
SPN_CTRL <- RunUMAP(SPN_CTRL, dims = 1:num_dims, reduction = "pca", reducti
on.name = "umap", reduction.key = "umap")
```

In [ ]:
```
#do plots
FeaturePlot(SPN_HD, reduction = "umap", features = "CAGLENGTH", cols = c("b
lue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_HD_clusters_cleaned.pdf", width = 8, height = 8)
FeaturePlot(SPN_HD, reduction = "umap", features = "EXP_CAGLENGTH", cols =
c("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_HD_clusters_cleaned_EXP_CAGLENGTH.pdf", width = 8, h
eight = 8)
DimPlot(SPN_HD, reduction = "umap", group.by = "PHASE", pt.size = 1)
ggsave("UMAP_Handsaker_HD_clusters_cleaned_PHASE.pdf", width = 8, height =
8)
DimPlot(SPN_HD, group.by = "SAMPLE", reduction = "umap", pt.size = 1)
ggsave("UMAP_Handsaker_HD_clusters_cleaned_splitBySampleMerged.pdf", width
= 8, height = 8)
DimPlot(SPN_HD, split.by = "SAMPLE", reduction = "umap", pt.size = 1, ncol
= 3, group.by = "SAMPLE") + NoLegend()
ggsave("UMAP_Handsaker_HD_clusters_cleaned_splitBySample.pdf", width = 20,
height = 20)
```

In [ ]:
```
#do plots
FeaturePlot(SPN_CTRL, reduction = "umap", features = "CAGLENGTH", cols = c
("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_CTRL_clusters_cleaned.pdf", width = 8, height = 8)
DimPlot(SPN_CTRL, reduction = "umap", group.by = "PHASE", cols = c("red"),
pt.size = 1)
ggsave("UMAP_Handsaker_CTRL_clusters_cleaned_PHASE.pdf", width = 8, height
= 8)
DimPlot(SPN_CTRL, group.by = "SAMPLE", reduction = "umap", pt.size = 1)
ggsave("UMAP_Handsaker_CTRL_clusters_cleaned_splitBySampleMerged.pdf", widt
h = 8, height = 8)
```

In [ ]:

```
#do plots for CAG size of HD samples
p <- ggplot(SPN_HD@meta.data, aes(x = CAGLENGTH)) +
geom_histogram(binwidth = 5, alpha = 0.5, position = "identity", fill = "#F
8766D") +
geom_vline(aes(xintercept = 150), colour="black", linetype = 2) +
  theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank())
plot(p)
ggsave("Handsaker_CAGLENGTH_hist.pdf", width = 8, height = 8)

p <- ggplot(SPN_HD@meta.data, aes(x = EXP_CAGLENGTH)) +
geom_histogram(binwidth = 5, alpha = 0.5, position = "identity", fill = "#F
8766D") +
geom_vline(aes(xintercept = 150), colour="black", linetype = 2) +
  theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank())
plot(p)
ggsave("Handsaker_CAGLENGTH_EXP_hist.pdf", width = 8, height = 8)
p <- ggplot(SPN_HD@meta.data, aes(x = EXP_CAGLENGTH, fill = SAMPLE)) +
geom_histogram(binwidth = 5, alpha = 0.5, position = "identity") +
geom_vline(aes(xintercept = 150), colour="black", linetype = 2) +
  theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank())
plot(p)
ggsave("Handsaker_CAGLENGTH_hist_splitBySample.pdf", width = 8, height = 8)

p <- ggplot(SPN_HD@meta.data, aes(x = EXP_CAGLENGTH, fill = SAMPLE)) +
geom_histogram(binwidth = 5, alpha = 0.5, position = "identity") +
geom_vline(aes(xintercept = 150), colour="black", linetype = 2) +
  theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank())
plot(p)
ggsave("Handsaker_CAGLENGTH_EXP_hist_splitBySample.pdf", width = 8, height
= 8)

p <- ggplot(SPN_HD@meta.data, aes(x = EXP_CAGLENGTH)) +
geom_histogram(binwidth = 5, alpha = 0.5, position = "identity", fill = "#F
8766D") +
xlim(c(150, 1000)) +
  theme(legend.title = element_text(size = 10),
```

```
        legend.text = element_text(size = 10),
        axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank())
plot(p)
ggsave("Handsaker_CAGLENGTH_hist_phaseC.pdf", width = 8, height = 8)

p <- ggplot(SPN_HD@meta.data, aes(x = EXP_CAGLENGTH, fill = SAMPLE)) +
geom_histogram(binwidth = 5, alpha = 0.5, position = "identity") +
xlim(c(150, 1000)) +
  theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank())
plot(p)
ggsave("Handsaker_CAGLENGTH_hist_phaseC_splitBySample.pdf", width = 8, heig
ht = 8)
```

Read_phaseC_genes

In [ ]:
```
phaseC_file <- "PROJECT_ELongATE/PhaseCD.txt"

phaseC_genes_all <- read.table(phaseC_file, sep = "\t", header = TRUE)
phaseC_genes <- phaseC_genes_all$Gene
```

In [ ]:
```
length(phaseC_genes)
```

In [ ]:
```
phaseC_plus_genes <- phaseC_genes_all[which(phaseC_genes_all$Phase.C.effect
> 0), "Gene"]
phaseC_minus_genes <- phaseC_genes_all[which(phaseC_genes_all$Phase.C.effec
t < 0), "Gene"]
```

In [ ]:
```
length(phaseC_plus_genes)
length(phaseC_minus_genes)
```

In [ ]:
```
#plot UMAP using only PhaseC genes as a test
SPN_HD <- RunPCA(SPN_HD, features = phaseC_genes, npcs = 50, reduction.name
= "pca_phaseC")
SPN_HD <- RunUMAP(SPN_HD, dims = 1:num_dims, reduction = "pca_phaseC", redu
ction.name = "umap_phaseC", reduction.key = "umap_phaseC")
FeaturePlot(SPN_HD, reduction = "umap_phaseC", features = "CAGLENGTH", cols
= c("blue", "red"), pt.size = 1)
ggsave("UMAP_Handsaker_HD_clusters_cleaned_phaseC_genesOnly_CAGLENGTH.pdf")
DimPlot(SPN_HD, reduction = "umap_phaseC", group.by = "PHASE", pt.size = 1)
ggsave("UMAP_Handsaker_HD_clusters_cleaned_phaseC_genesOnly_PHASE.pdf")
```

Learn_phase_model

In [ ]:
```r
#extract phase C genes that are expressed in the dataset
phaseC_genes <- intersect(phaseC_genes, rownames(SPN_HD@assays$SCT$data))
phaseC_plus_genes <- intersect(phaseC_plus_genes, phaseC_genes)
phaseC_minus_genes <- intersect(phaseC_minus_genes, phaseC_genes)

#extract SPNs with a phase (i.e. > 36 CAG, otherwise PHASE is NA)
PHASE_SPN <- SPN_HD@meta.data[, "PHASE"]

#extract counts for Phase C genes
phaseC_counts_SPN_phased <- SPN_HD@assays$SCT$data[phaseC_genes, which(!is.
na(PHASE_SPN))]
phaseC_counts_SPN <- SPN_HD@assays$SCT$data[phaseC_genes, ]

#extract training (70%) and test (30%) data for phase model
training_test_data_phase <- data.frame(PHASE = PHASE_SPN, t(phaseC_counts_S
PN))
set.seed(1)
ind_training_phase <- sample(x = 1:round(length(PHASE_SPN)), size = round(l
ength(PHASE_SPN)*0.7), replace = FALSE)
ind_test_phase <- setdiff(1:round(length(PHASE_SPN)), ind_training_phase)

training_data_phase <- training_test_data_phase[ind_training_phase, ]
test_data_phase <- training_test_data_phase[ind_test_phase, ]

training_data_phase_noNA <- training_data_phase[which(!is.na(training_data_
phase$PHASE)), ]
test_data_phase_noNA <- test_data_phase[which(!is.na(test_data_phase$PHAS
E)), ]

phase_training <- training_data_phase[, -1]
phase_training_noNA <- training_data_phase_noNA[, -1]
phase_test <- test_data_phase[, -1]
phase_test_noNA <- test_data_phase_noNA[, -1]

phase_all_HD <- training_test_data_phase[, -1]
phase_all_HD_noNA <- rbind(training_data_phase_noNA, test_data_phase_noNA)
[, -1]

phaseC_genes_CTRL <- intersect(phaseC_genes, rownames(SPN_CTRL@assays$SCT$d
ata))
PHASE_SPN_CTRL <- SPN_CTRL@meta.data[, "PHASE"]
phaseC_counts_SPN_CTRL <- SPN_CTRL@assays$SCT$data[phaseC_genes, ]

test_data_phase_CTRL <- data.frame(PHASE = PHASE_SPN_CTRL, t(phaseC_counts_
SPN_CTRL))
phase_all_CTRL <- test_data_phase_CTRL[, -1]
```

In [ ]:
```r
#setdiff(colnames(training_test_data_phase), phaseC_genes)
#setdiff(phaseC_genes, colnames(training_test_data_phase))
```

In [ ]:
```r
dim(phase_training_noNA)
dim(phase_test_noNA)
dim(phase_all_HD)
dim(phase_all_HD_noNA)
dim(phase_all_CTRL)
```

In [ ]:
```
#learn phase model with glmnet
model_phase <- cv.glmnet(as.matrix(phase_training_noNA),
                         factor(training_data_phase_noNA[, 1], levels = c
("A-B", "C-D-E")),
                         family = "binomial",
                         type.measure = "class")
```

In [ ]:
```
# saveRDS(model_phase, "Phase_model.rds")
```

In [ ]:

```
class_thr <- 0.5
lambda_val <- "lambda.min"

#run phase model on training set
predicted_phase_training_noNA_wprob <- predict(model_phase, newx = as.matri
x(phase_training_noNA), s = lambda_val, type = "response") #glmnet
predicted_phase_training_noNA <- rep("A-B", dim(predicted_phase_training_no
NA_wprob)[1])
predicted_phase_training_noNA[which(predicted_phase_training_noNA_wprob > c
lass_thr)] <- "C-D-E" #glmnet
predicted_phase_training_wprob <- predict(model_phase, newx = as.matrix(pha
se_training), s = lambda_val, type = "response") #glmnet
predicted_phase_training <- rep("A-B", dim(predicted_phase_training_wprob)
[1])
predicted_phase_training[which(predicted_phase_training_wprob > class_thr)]
<- "C-D-E" #glmnet
res_training <- data.frame(pred = predicted_phase_training_noNA_wprob, pred
icted_phase_training_noNA, PHASE = training_data_phase_noNA[, 1])
print(table(res_training[, c(2, 3)])) #glmnet
acc_training <- (table(res_training[, c(2, 3)])[1, 1] + table(res_training
[, c(2, 3)])[2, 2])/(sum(table(res_training[, c(2, 3)]))) #glmnet

#run phase model on test set
predicted_phase_test_noNA_wprob <- predict(model_phase, newx = as.matrix(ph
ase_test_noNA), s = lambda_val, type = "response") #glmnet
predicted_phase_test_noNA <- rep("A-B", dim(predicted_phase_test_noNA_wpro
b)[1])
predicted_phase_test_noNA[which(predicted_phase_test_noNA_wprob > class_th
r)] <- "C-D-E" #glment
predicted_phase_test_wprob <- predict(model_phase, newx = as.matrix(phase_t
est), s = lambda_val, type = "response") #glmnet
predicted_phase_test <- rep("A-B", dim(predicted_phase_test_wprob)[1])
predicted_phase_test[which(predicted_phase_test_wprob > class_thr)] <- "C-D
-E" #glment
res_test <- data.frame(pred = predicted_phase_test_noNA_wprob, predicted_ph
ase_test_noNA, PHASE = test_data_phase_noNA[, 1])
print(table(res_test[, c(2, 3)])) #glmnet
acc_test <- (table(res_test[, c(2, 3)])[1, 1] + table(res_test[, c(2, 3)])
[2, 2])/(sum(table(res_test[, c(2, 3)]))) #glmnet

#run phase model on all HD SPN
predicted_phase_all_HD_wprob <- predict(model_phase, newx = as.matrix(phase
_all_HD), s = lambda_val, type = "response") #glmnet
predicted_phase_all_HD <- rep("A-B", dim(predicted_phase_all_HD_wprob)[1])
predicted_phase_all_HD[which(predicted_phase_all_HD_wprob > class_thr)] <-
"C-D-E" #glmnet
predicted_phase_all_HD_noNA_wprob <- predict(model_phase, newx = as.matrix
(phase_all_HD_noNA), s = lambda_val, type = "response") #glmnet
predicted_phase_all_HD_noNA <- rep("A-B", dim(predicted_phase_all_HD_noNA_w
prob)[1])
predicted_phase_all_HD_noNA[which(predicted_phase_all_HD_noNA_wprob > class
_thr)] <- "C-D-E" #glmnet

#run phase model on all CTRL SPN
predicted_phase_all_CTRL_wprob <- predict(model_phase, newx = as.matrix(pha
se_all_CTRL), s = lambda_val, type = "response") #glmnet
predicted_phase_all_CTRL <- rep("A-B", dim(predicted_phase_all_CTRL_wprob)
[1])
predicted_phase_all_CTRL[which(predicted_phase_all_CTRL_wprob > class_thr)]
<- "C-D-E" #glmnet
```

```
cat(sprintf("\nPhase model accuracy on training set: %.3f\n", acc_trainin
g))
cat(sprintf("Phase model accuracy on test set: %.3f\n", acc_test))
```

In [ ]:
```
table(predicted_phase_training)
table(predicted_phase_training_noNA)
table(predicted_phase_test)
table(predicted_phase_test_noNA)
table(predicted_phase_all_CTRL)
```

In [ ]:
```
#extract genes selected by the phase model
tmp_coef_phase <- coef(model_phase, s = lambda_val)
selected_features_phase <- names(tmp_coef_phase[tmp_coef_phase@i + 1, ])[-
1]
length(selected_features_phase)
```

In [ ]:
```
#head(res_training)
length(which(res_training$predicted_phase == "A-B"))
length(which(res_training$predicted_phase == "C-D-E"))
length(which(res_training$PHASE == "A-B"))
length(which(res_training$PHASE == "C-D-E"))
```

In [ ]:
```
#add predicted phase to HD samples and plot UMAP
SPN_HD@meta.data[, "PREDICTED_PHASE"] <- predicted_phase_all_HD
DimPlot(SPN_HD, reduction = "umap", group.by = "PREDICTED_PHASE", pt.size =
1)
ggsave("UMAP_Handsaker_HD_clusters_cleaned_PREDICTED_PHASE.pdf", width = 8,
height = 8)

#add predicted phase to CTRL sample and plot UMAP
SPN_CTRL@meta.data[, "PREDICTED_PHASE"] <- predicted_phase_all_CTRL
DimPlot(SPN_CTRL, reduction = "umap", group.by = "PREDICTED_PHASE", pt.size
= 1)
ggsave("UMAP_Handsaker_CTRL_clusters_cleaned_PREDICTED_PHASE.pdf", width =
8, height = 8)
```

In [ ]:
```
SPN_all <- merge(SPN_HD, SPN_CTRL)
```

In [ ]:
```
head(SPN_all)
```

Define training and test set for CAG sizing model

In [ ]:
```
CAGLENGTH_SPN_sized <- SPN_HD@meta.data[, "CAGLENGTH"]
training_test_data <- data.frame(cbind(CAGLENGTH_SPN_sized, t(phaseC_counts
_SPN)))

set.seed(1)
ind_training <- sample(x = 1:round(dim(SPN_HD@assays$SCT$data)[2]), size =
round(dim(SPN_HD@assays$SCT$data)[2]*0.7), replace = FALSE)
ind_test <- setdiff(1:round(dim(SPN_HD@assays$SCT$data)[2]), ind_training)

training_data <- training_test_data[ind_training, ]
test_data <- training_test_data[ind_test, ]
```

In [ ]:
```
length(CAGLENGTH_SPN_sized)
dim(training_test_data_phase)
```

In [ ]:
```
head(phaseC_counts_SPN)
#apply(phaseC_counts_SPN, 1, mean)
```

In [ ]:
```
training_data
test_data
```

In [ ]:
```
dim(training_test_data)
```

In [ ]:
```
summary(CAGLENGTH_SPN_sized)
length(sort(CAGLENGTH_SPN_sized))
head(training_data)
```

In [ ]:
```
Handsaker_donor_metadata <- read.table("PROJECT_ELongATE/Handsaker/donor_me
tadata.txt", header = TRUE)
Handsaker_donor_metadata <- Handsaker_donor_metadata[which(Handsaker_donor_
metadata$SID %in% unique(SPN_all$DONOR)), ]
rownames(Handsaker_donor_metadata) <- Handsaker_donor_metadata$SID
Handsaker_donor_metadata
```

In [ ]:
```
tmp_Handsaker <- do.call(rbind,lapply(split(SPN_HD@meta.data$PHASE, SPN_HD@
meta.data$SAMPLE), function(x) {
    tab <- table(x)
}))

#tmp_Handsaker
HD_Handsaker_fract_CDE <- data.frame(DATASET = "Handsaker_Caudate", SAMPLE
= rownames(tmp_Handsaker), FRACT_CDE=100*tmp_Handsaker[, 2]/(tmp_Handsaker
[, 1] + tmp_Handsaker[, 2]), NUM_SPN=tmp_Handsaker[, 1] + tmp_Handsaker[,
2])
HD_Handsaker_fract_CDE$GRADE <- Handsaker_donor_metadata[rownames(HD_Handsa
ker_fract_CDE), "VS_Grade"]
HD_Handsaker_fract_CDE$FRACT_SPN <- num_cells_Handsaker[rownames(HD_Handsak
er_fract_CDE), "num_spns"]/num_cells_Handsaker[rownames(HD_Handsaker_fract_
CDE), "num_cells"]
HD_Handsaker_fract_CDE
```

In [ ]:
```
ggplot(HD_Handsaker_fract_CDE, aes(x = GRADE, y = FRACT_CDE, size = FRACT_S
PN)) +
geom_point(aes(size = FRACT_SPN), color = "#F8766D", alpha = 0.8, position
= position_jitter(width = 0, height = 0)) +
#scale_color_manual(values =c("#F8766D")) +
theme_classic() +
xlab("HD Grade") +
ylim(c(0, 25)) +
ylab("Fraction cells in C-D-E phase (%)")
ggsave("HD_Handsaker_fraction_CDE.pdf", width = 6, height = 6)
```

In [ ]:

```
#add metadata
tmp <- lapply(split(SPN_HD$PHASE, SPN_HD$DONOR), function(x) {val <- table
(x)[2]/(table(x)[1] + table(x)[2]); names(val) <- "FRACT_CDE"; return(va
l)})
tmp2 <- do.call(rbind, tmp)
tmp3 <- lapply(split(SPN_HD$PREDICTED_PHASE, SPN_HD$DONOR), function(x) {va
l <- table(x)[2]/(table(x)[1] + table(x)[2]); names(val) <- "PRED_FRACT_CD
E"; return(val)})
tmp4 <- do.call(rbind, tmp3)
tmp5 <- lapply(split(SPN_HD@meta.data[ind_test, "PHASE"], SPN_HD@meta.data
[ind_test, "DONOR"]), function(x) {val <- table(x)[2]/(table(x)[1] + table
(x)[2]); names(val) <- "FRACT_CDE_TEST"; return(val)})
tmp6 <- do.call(rbind, tmp5)
tmp7 <- lapply(split(SPN_HD@meta.data[ind_test, "PREDICTED_PHASE"], SPN_HD@
meta.data[ind_test, "DONOR"]), function(x) {
    if (length(unique(x)) > 1) {
        val <- table(x)[2]/(table(x)[1] + table(x)[2])
    } else {
        val <- 0
    }
    names(val) <- "PRED_FRACT_CDE_TEST"; return(val)})
tmp8 <- do.call(rbind, tmp7)
tmp9 <- lapply(split(SPN_HD@meta.data[ind_test, ], SPN_HD@meta.data[ind_tes
t, "DONOR"]), function(x) {val <- dim(x)[1]; names(val) <- "NUM_SPN_TEST";
return(val)})
tmp10 <- do.call(rbind, tmp9)
tmp11 <- lapply(split(SPN_HD@meta.data[, ], SPN_HD@meta.data[, "DONOR"]), f
unction(x) {val <- dim(x)[1]; names(val) <- "NUM_SPN"; return(val)})
tmp12 <- do.call(rbind, tmp11)

Handsaker_donor_metadata_HD <- Handsaker_donor_metadata[grep(Handsaker_dono
r_metadata$Status, pattern = "Control", invert = TRUE), ]
GERM_CAGEXP <- as.numeric(gsub(x = Handsaker_donor_metadata_HD$CAG, pattern
= ".*/", replacement = ""))

Handsaker_donor_metadata_HD <- cbind(tmp2, tmp4, tmp6, tmp8, tmp10, tmp12,
GERM_CAGEXP, Handsaker_donor_metadata_HD[rownames(tmp2), ])

Handsaker_donor_metadata_HD$CAP <- as.numeric(Handsaker_donor_metadata_HD$C
AP)
Handsaker_donor_metadata_HD$FRACT_CDE_TEST[which(is.na(Handsaker_donor_meta
data_HD$FRACT_CDE_TEST))] <- 0
Handsaker_donor_metadata_HD$FRACT_CDE[which(is.na(Handsaker_donor_metadata_
HD$FRACT_CDE))] <- 0
Handsaker_donor_metadata_HD
Handsaker_donor_metadata_CTRL <- Handsaker_donor_metadata[grep(Handsaker_do
nor_metadata$Status, pattern = "Control", invert = FALSE), ]

tmpC <- lapply(split(SPN_CTRL$PHASE, SPN_CTRL$DONOR), function(x) {val <- t
able(x)[2]/(table(x)[1] + table(x)[2]); names(val) <- "FRACT_CDE"; return(v
al)})
tmp2C <- do.call(rbind, tmpC)

tmp3C <- lapply(split(SPN_CTRL$PREDICTED_PHASE, SPN_CTRL$DONOR), function
(x) {val <- table(x)[2]/(table(x)[1] + table(x)[2]); names(val) <- "PRED_FR
ACT_CDE"; return(val)})
tmp4C <- do.call(rbind, tmp3C)
tmp5C <- lapply(split(SPN_CTRL@meta.data[, "PHASE"], SPN_CTRL@meta.data[,
"DONOR"]), function(x) {val <- table(x)[2]/(table(x)[1] + table(x)[2]); nam
es(val) <- "FRACT_CDE_TEST"; return(val)})
```

```
tmp6C <- do.call(rbind, tmp5C)
tmp7C <- lapply(split(SPN_CTRL@meta.data[, "PREDICTED_PHASE"], SPN_CTRL@met
a.data[, "DONOR"]), function(x) {
    if (length(unique(x)) > 1) {
        val <- table(x)[2]/(table(x)[1] + table(x)[2])
    } else {
        val <- 0
    }
    names(val) <- "PRED_FRACT_CDE_TEST"; return(val)})
tmp8C <- do.call(rbind, tmp7C)
tmp9C <- lapply(split(SPN_CTRL@meta.data, SPN_CTRL@meta.data[, "DONOR"]), f
unction(x) {val <- dim(x)[1]; names(val) <- "NUM_SPN_TEST"; return(val)})
tmp10C <- do.call(rbind, tmp9C)
tmp11C <- lapply(split(SPN_CTRL@meta.data[, ], SPN_CTRL@meta.data[, "DONO
R"]), function(x) {val <- dim(x)[1]; names(val) <- "NUM_SPN"; return(val)})
tmp12C <- do.call(rbind, tmp11C)

Handsaker_donor_metadata_CTRL <- cbind(tmp2C, tmp4C, tmp6C, tmp8C, tmp10C,
tmp12C, GERM_CAGEXP = NA, Handsaker_donor_metadata_CTRL[rownames(tmp4C), ])
Handsaker_donor_metadata_CTRL$FRACT_CDE <- 0
Handsaker_donor_metadata_CTRL$PRED_FRACT_CDE <- 0
Handsaker_donor_metadata_CTRL
```

In [ ]:
```
Handsaker_donor_metadata_full <- rbind(Handsaker_donor_metadata_HD, Handsak
er_donor_metadata_CTRL)
```

In [ ]:
```
#plot fraction of SPN in CDE phase in the test set
cor_predFractCDE_fractCDE <- weighted.cor(x = Handsaker_donor_metadata_full
$FRACT_CDE_TEST, y = Handsaker_donor_metadata_full$PRED_FRACT_CDE_TEST, wei
ghts = Handsaker_donor_metadata_full$NUM_SPN_TEST, method = "pearson", na.r
m=TRUE)
ggplot(Handsaker_donor_metadata_full, aes(x=100*FRACT_CDE_TEST, y=100*PRED_
FRACT_CDE_TEST, size = NUM_SPN_TEST)) +
  geom_point(color = "blue") +
  geom_smooth(method=lm, se=TRUE, mapping = aes(weight = NUM_SPN_TEST)) +
  theme_classic() +
  xlab("Fraction cells in C-D-E phase (%)") +
  ylab("Predicted fraction cells in C-D-E phase (%)")

sprintf("Sp. corr = %.2f", cor_predFractCDE_fractCDE)
ggsave("Handsaker_fraction_SPN_CDE_phase_test_set.pdf", width = 8, height =
8)
```

In [ ]:
```
#plot fraction of SPN in CDE phase in the test set vs CAP score
cor_CAP_fractCDE <- weighted.cor(x = Handsaker_donor_metadata_HD$FRACT_CDE,
y = Handsaker_donor_metadata_HD$CAP, weights = Handsaker_donor_metadata_HD
$NUM_SPN, method = "pearson", na.rm=TRUE)
ggplot(Handsaker_donor_metadata_HD, aes(x=CAP, y=100*FRACT_CDE, size = NUM_
SPN)) +
  geom_point(color = "blue") +
  geom_smooth(method=lm, se=TRUE, mapping = aes(weight = NUM_SPN)) +
  theme_classic() +
  xlab("CAP score") +
  ylab("Fraction cells in C-D-E phase (%)")
sprintf("Sp. corr = %.2f", cor_CAP_fractCDE)
ggsave("Handsaker_fraction_SPN_CDE_phase_CAP.pdf", width = 8, height = 8)
```

In [ ]:
```
str(Handsaker_donor_metadata_HD)
str(Handsaker_donor_metadata_full)
```

In [ ]:
```
#plot fraction of SPN in CDE phase in the test set vs num. CAG in the germl
ine
cor_numCAGGerm_fractCDE <- weighted.cor(x = Handsaker_donor_metadata_HD$GER
M_CAGEXP, y = Handsaker_donor_metadata_HD$FRACT_CDE, weights = Handsaker_do
nor_metadata_HD$NUM_SPN, method = "pearson", na.rm=TRUE)

ggplot(Handsaker_donor_metadata_HD, aes(x=GERM_CAGEXP, y=100*FRACT_CDE, siz
e = NUM_SPN)) +
  geom_point(color = "blue") +
  geom_smooth(method=lm, se=TRUE, mapping = aes(weight = NUM_SPN)) +
  theme_classic() +
   xlab("Num. CAG germline") +
  ylab("Fraction cells in C-D-E phase (%)")

print(sprintf("Sp. corr = %.2f", cor_numCAGGerm_fractCDE))
ggsave("Handsaker_numCAGGermline_vs_fraction_SPN_CDE.pdf", width = 8, heigh
t = 8)
```

In [ ]:
```
#wtd.cor(x = Handsaker_donor_metadata_HD$GERM_CAGEXP, y = Handsaker_donor_m
etadata_HD$FRACT_CDE, weight = Handsaker_donor_metadata_HD$NUM_SPN)
#wtd.cor(x = Handsaker_donor_metadata_HD$CAP, y = Handsaker_donor_metadata_
HD$FRACT_CDE, weight = Handsaker_donor_metadata_HD$NUM_SPN)
```

Learn_CAG_model

In [ ]:
```
#extract filtered training set with CAG > CAG_thr
expr_C_training <- training_data[, -1]
CAG_THR <- 150

training_data_filtered <- training_data[which(training_data$CAGLENGTH_SPN_s
ized > CAG_THR), ]
expr_C_training_filtered <- training_data_filtered[, -1]

#learn CAG sizing model
fit_glm <- cv.glmnet(as.matrix(expr_C_training_filtered), as.vector(trainin
g_data_filtered[, 1]))
```

In [ ]:
```
saveRDS(fit_glm, "CAG_sizing_model.rds")
```

In [ ]:
```
#dim(training_data[, -1])
dim(expr_C_training)
dim(expr_C_training_filtered)
```

In [ ]:
```
plot(fit_glm)
head(coef(fit_glm, s = lambda_val))
#coef(fit_glm) #same as s = "lambda.1se"
coef(fit_glm, s = lambda_val)
```

Predict_on_training_set

In [ ]:
```
class_thr <- 0.5
lambda_val <- "lambda.min"

#predict CAG size on filtered training set
pred_training_filtered <- predict(fit_glm, newx = as.matrix(expr_C_training
_filtered), s = lambda_val) #glmnet
#predict phase on filtered training set
pred_training_filtered_phase_wprob <- predict(model_phase, newx = as.matrix
(expr_C_training_filtered), s = lambda_val, type = "response") #glmnet
pred_training_filtered_phase <- rep("A-B", dim(pred_training_filtered_phase
_wprob)[1])
pred_training_filtered_phase[which(pred_training_filtered_phase_wprob > cla
ss_thr)] <- "C-D-E" #glmnet
#predict CAG size on training set
pred_training <- predict(fit_glm, newx = as.matrix(expr_C_training), s = la
mbda_val)
#predict phase on training set
pred_training_phase_wprob <- predict(model_phase, newx = as.matrix(expr_C_t
raining), s = lambda_val, type = "response") #glmnet
pred_training_phase <- rep("A-B", dim(pred_training_phase_wprob)[1])
pred_training_phase[which(pred_training_phase_wprob > class_thr)] <- "C-D-
E" #glmnet

#extract expression of phase C genes for test set
expr_C_test <- test_data[, -1]
#predict CAG size on test set
pred_test <- predict(fit_glm, newx = as.matrix(expr_C_test), s = lambda_va
l) #glmnet
#predict phase on test set
pred_test_phase_wprob <- predict(model_phase, newx = as.matrix(expr_C_tes
t), s = lambda_val, type = "response") #glmnet
pred_test_phase <- rep("A-B", dim(pred_test_phase_wprob)[1])
pred_test_phase[which(pred_test_phase_wprob > class_thr)] <- "C-D-E" #glmne
t
```

In [ ]:
```
tmp_coef <- coef(fit_glm, s = lambda_val)
selected_features <- names(tmp_coef[tmp_coef@i + 1, ])[-1]
```

In [ ]:
```
selected_features
length(selected_features) #79
```

In [ ]:
```
#training_data_filtered_sorted
coef(fit_glm, s = lambda_val)[, 1][selected_features]
```

In [ ]:

```
#sort SPNs of filtered training set by CAG length
training_data_filtered_sorted <- training_data_filtered[order(training_data
_filtered$CAGLENGTH_SPN_sized), ]
head(training_data_filtered_sorted)
#plot(training_data_filtered_sorted[, 1], apply(training_data_filtered_sort
ed[, intersect(colnames(training_data_filtered_sorted), phaseC_plus_gene
s)], 1, mean))
#plot(training_data_filtered_sorted[, 1], apply(training_data_filtered_sort
ed[, intersect(colnames(training_data_filtered_sorted), phaseC_minus_gene
s)], 1, mean))

#plot average expression of all phase C+ and phase C- genes
plot(training_data_filtered_sorted[, 1], apply(training_data_filtered_sorte
d[, intersect(intersect(colnames(training_data_filtered_sorted), phaseC_plu
s_genes), selected_features)], 1, mean))
plot(training_data_filtered_sorted[, 1], apply(training_data_filtered_sorte
d[, intersect(intersect(colnames(training_data_filtered_sorted), phaseC_min
us_genes), selected_features)], 1, mean))
```

In [ ]:

```
#extract features selected by CAG sizing model and associated coefficients
selected_features_plus <-  intersect(intersect(colnames(training_data_filte
red_sorted), phaseC_plus_genes), selected_features)
selected_features_minus <-  intersect(intersect(colnames(training_data_filt
ered_sorted), phaseC_minus_genes), selected_features)
coef_selected_features_plus <- coef(fit_glm, s = lambda_val)[, 1][selected_
features_plus]
coef_selected_features_minus <- coef(fit_glm, s = lambda_val)[, 1][selected
_features_minus]
df_plus_training_filtered <- data.frame(CAGLENGTH_SPN_sized=training_data_f
iltered_sorted[, "CAGLENGTH_SPN_sized"], score = apply(mapply(function(x,
y) x*y, training_data_filtered_sorted[, selected_features_plus], coef_selec
ted_features_plus), 1, function(x) sum(x)) + coef(fit_glm, s = lambda_val)
[1, 1]/2, signature = "plus")
df_minus_training_filtered <- data.frame(CAGLENGTH_SPN_sized=training_data_
filtered_sorted[, "CAGLENGTH_SPN_sized"], score = apply(mapply(function(x,
y) x*y, training_data_filtered_sorted[, selected_features_minus], coef_sele
cted_features_minus), 1, function(x) sum(x) + coef(fit_glm, s = lambda_val)
[1, 1]/2), signature = "minus")

#extract features selected by CAG sizing model and obtain score for SPNs fr
om test set
test_data_HD <- cbind(test_data, PROB_PHASE_CDE=pred_test_phase_wprob[, 1])
test_data_HD <- test_data_HD[which(test_data_HD$CAGLENGTH_SPN_sized > 36),
]
df_plus <- data.frame(CAGLENGTH_SPN_sized=test_data_HD[, "CAGLENGTH_SPN_siz
ed"], score = apply(mapply(function(x, y) x*y, test_data_HD[, selected_feat
ures_plus], coef_selected_features_plus), 1, function(x) sum(x)) + coef(fit
_glm, s = lambda_val)[1, 1]/2, signature = "plus")
df_minus <- data.frame(CAGLENGTH_SPN_sized=test_data_HD[, "CAGLENGTH_SPN_si
zed"], score = apply(mapply(function(x, y) x*y, test_data_HD[, selected_fea
tures_minus], coef_selected_features_minus), 1, function(x) sum(x)) + coef
(fit_glm, s = lambda_val)[1, 1]/2, signature = "minus")
#extract all Phase C+ and Phase C- genes and obtain C+ and C- scores for SP
Ns from test set
df_plus_allPhaseCgenes <- data.frame(CAGLENGTH_SPN_sized=test_data_HD[, "CA
GLENGTH_SPN_sized"], score = apply(test_data_HD[, gsub(x = phaseC_plus_gene
s, pattern = "-", replacement = ".")], 1, function(x) mean(x)), signature =
"plus")
df_minus_allPhaseCgenes <- data.frame(CAGLENGTH_SPN_sized=test_data_HD[, "C
AGLENGTH_SPN_sized"], score = apply(test_data_HD[, gsub(x = phaseC_minus_ge
nes, pattern = "-", replacement = ".")], 1, function(x) mean(x)), signature
= "minus")
```

In [ ]:

```
#plot scores
df <- data.frame(CAGLENGTH_SPN_sized=test_data_HD[, "CAGLENGTH_SPN_sized"],
score = df_plus$score + df_minus$score, PROB_PHASE_CDE = test_data_HD[, "PR
OB_PHASE_CDE"])
df_sorted <- df[order(df$CAGLENGTH_SPN_sized), ]
df_sorted$PHASE <- "A-B"
df_sorted$PHASE[which(df_sorted$CAGLENGTH_SPN_sized > 150)] <- "C-D-E"
df_sorted$PRED_PHASE <- "A-B"
df_sorted$PRED_PHASE[which(df_sorted$PROB_PHASE_CDE > 0.5)] <- "C-D-E"
df_sorted$index <- seq_along(df_sorted$score)

df_allPhaseCgenes <- data.frame(CAGLENGTH_SPN_sized=test_data_HD[, "CAGLENG
TH_SPN_sized"], avg_expr_PhaseC = c(df_minus_allPhaseCgenes$score, df_plus_
allPhaseCgenes$score))
df_allPhaseCgenes_sorted <- df_allPhaseCgenes[order(df_allPhaseCgenes$CAGLE
NGTH_SPN_sized), ]
df_allPhaseCgenes_sorted$PHASE <- "A-B"
df_allPhaseCgenes_sorted$PHASE[which(df_allPhaseCgenes_sorted$CAGLENGTH_SPN
_sized > 150)] <- "C-D-E"
df_allPhaseCgenes_sorted$index <- seq_along(df_allPhaseCgenes_sorted$avg_ex
pr_PhaseC)

p1 <- ggplot(df_allPhaseCgenes_sorted, aes(x = index, y = avg_expr_PhaseC,
color = PHASE)) +
geom_point(aes(x = index, y = avg_expr_PhaseC, color = PHASE), size = 1) +
theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        #axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.x = element_blank(),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank()) +
xlab("SPNs sorted by CAG length") +
ylab("Average expression Phase C genes")

ind_150 <- df_sorted[which(df_sorted$CAGLENGTH_SPN_sized > 150), "index"]
[1]
p2 <- ggplot(df_sorted, aes(x = index)) +
geom_point(aes(y = score, color = PRED_PHASE, alpha = PROB_PHASE_CDE), size
= 1) +
geom_point(aes(y = CAGLENGTH_SPN_sized), size = 0.1) +
theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        #axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.x = element_blank(),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank()) +
geom_vline(aes(xintercept = ind_150), colour="black", linetype = 2) +
#geom_hline(aes(yintercept = 150), colour="black", linetype = 2) +
xlab("SPNs sorted by CAG length") +
ylab("Number of CAG repeats")

df_sorted_filtered <- df_sorted[which(df_sorted$CAGLENGTH_SPN_sized > 150 |
df_sorted$PROB_PHASE_CDE > 0.5), ]
ind_150_filt <- df_sorted_filtered[which(df_sorted_filtered$CAGLENGTH_SPN_s
ized > 150), "index"][1]
p3 <- ggplot(df_sorted_filtered, aes(x = index)) +
```

```
geom_point(aes(y = score, color = PRED_PHASE, alpha = PROB_PHASE_CDE), size
= 1) +
geom_point(aes(y = CAGLENGTH_SPN_sized), size = 0.1) +
theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        #axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.x = element_blank(),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank()) +
xlim(c(ind_150_filt, max(df_sorted_filtered$index))) +
xlab("SPNs sorted by CAG length") +
ylab("Number of CAG repeats")

#ggarrange(p1, p2, nrow = 2, ncol = 1)

plot(p1)
ggsave("Handsaker_HD_PhaseC_expression.pdf", height = 8, width = 8)
plot(p2)
ggsave("Handsaker_HD_CAG_sizing_model.pdf", height = 8, width = 8)
plot(p3)
ggsave("Handsaker_HD_CAG_sizing_model_PhaseC_only.pdf", height = 8, width =
8)
```

In [ ]:
```
#find number of phase C genes expressed in each SPN
num_expr_genes_training <- apply(as.matrix(expr_C_training[, selected_featu
res]), 1, function(x) length(which(x > 0)))
num_expr_genes_training_filtered <- apply(as.matrix(expr_C_training_filtere
d[, selected_features]), 1, function(x) length(which(x > 0)))
num_expr_genes_test <- apply(as.matrix(expr_C_test[, selected_features]),
1, function(x) length(which(x > 0)))

summary(num_expr_genes_training)
summary(num_expr_genes_training_filtered)
summary(num_expr_genes_test)
```

In [ ]:
```
head(pred_training_filtered)
head(pred_training_filtered_phase)
```

In [ ]:
```
dim(pred_training)
dim(pred_training_filtered)
dim(pred_test)
dim(training_data)
```

In [ ]:

```
#create dataframes with model predictions
results_training_filtered <- data.frame(CAG_measured = training_data_filter
ed[, 1], CAG_predicted = as.numeric(pred_training_filtered), pred_phase = p
red_training_filtered_phase, num_expr_genes = num_expr_genes_training_filte
red)
results_training <- data.frame(CAG_measured = training_data[, 1], CAG_predi
cted = as.numeric(pred_training), pred_phase = pred_training_phase, num_exp
r_genes = num_expr_genes_training)
CAG_measured_exp <- training_data[, 1]
CAG_measured_exp[which(CAG_measured_exp < 36)] <- NA

results_training_exp <- data.frame(CAG_measured = CAG_measured_exp, CAG_pre
dicted = as.numeric(pred_training), pred_phase = pred_training_phase, num_e
xpr_genes = num_expr_genes_training)
results_training_phaseC <- results_training[which(results_training$pred_pha
se == "C-D-E"), ]
results_training_phaseC_exp <- results_training_exp[which(results_training_
exp$pred_phase == "C-D-E"), ]
results_training_phaseC_measured <- results_training[which(results_training
$CAG_measured > 150), ]

#plot correlation between CAG sizing measured values and predictions
Evaluate_correlation <- function(Data, x, y, quant_thr = 1, abl = TRUE, not
es = "") {
  correlation <- cor(Data[, x], Data[, y], method = "pearson", use = "compl
ete.obs")
  pdf(paste0("AccuracyPred_", notes, ".pdf"))
  smoothScatter(Data[, x], Data[, y], xlab = x, ylab = y, cex.main = 0.6, c
ex.axis = 0.8,
                xlim = c(0, quantile(c(Data[, x], Data[, y]), quant_thr, n
a.rm = TRUE)),
                ylim = c(0, quantile(c(Data[, x], Data[, y]), quant_thr, n
a.rm = TRUE)))

  if (abl) {
    abline(0, 1, col = "green")
  }
  dev.off()

  smoothScatter(Data[, x], Data[, y], xlab = x, main = paste0(y, " VS ", x,
"\n r Pearson. = ",
                sprintf("%.2f", correlation), "\n ", notes, " - n = ", dim
(Data)[1]), ylab = y, cex.main = 0.6, cex.axis = 0.8,
                xlim = c(0, quantile(c(Data[, x], Data[, y]), quant_thr, n
a.rm = TRUE)), ylim = c(0, quantile(c(Data[, x], Data[, y]), quant_thr, na.
rm = TRUE)))

  if (abl) {
    abline(0, 1, col = "green")
  }

  print(paste0(y, " VS ", x, "\n r Pearson. = ",
               sprintf("%.2f", correlation), "\n ", notes, " - n = ", dim(Da
ta)[1]))


  ind_noNA <- which(apply(Data, 1, function(x) !any(is.na(x))))
  Data_filt <- Data[ind_noNA, ]
  df <- data.frame(CAG = c(Data_filt[, x], Data_filt[, y]), method = c(rep
("Measured", dim(Data_filt)[1]), rep("Predicted", dim(Data_filt)[1])))
```

```
  p <- ggplot(df, aes(x= CAG, fill = method)) +
  geom_histogram(binwidth = 5, alpha = 0.5, position = "identity") +
  xlab("Num. CAG") + ylab("Num. SPNs") +
  theme(legend.title = element_text(size = 10),
        legend.text = element_text(size = 10),
        axis.text.x = element_text(size = 10, angle = 90, vjust = 0.5),
        axis.text.y = element_text(size = 10),
        axis.title.x = element_text(size = 14),
        axis.title.y = element_text(size = 14),
        panel.background = element_blank())

print(gsub(x = notes, pattern = "_", replacement = " "))
  #  +  geom_vline(aes(xintercept = 150), colour="black", linetype = 2)

  plot(p)

  ggsave(paste0("AccuracyPred_hist_", notes, ".pdf"), width = 8, height =
8)
}

Evaluate_correlation(results_training_filtered, "CAG_measured", "CAG_predic
ted", notes = "Training_set_Filtered_SPN", quant_thr = 0.999)
#Evaluate_correlation(results_training, "CAG_measured", "CAG_predicted", no
tes = "Training_set_All_SPN", quant_thr = 0.999)
Evaluate_correlation(results_training_phaseC, "CAG_measured", "CAG_predicte
d", notes = "Training_set_Phase_C_SPN", quant_thr = 0.999)
Evaluate_correlation(results_training_phaseC_exp, "CAG_measured", "CAG_pred
icted", notes = "Training_set_Phase_C_SPN_expOnly", quant_thr = 0.999)
Evaluate_correlation(results_training_phaseC_measured, "CAG_measured", "CAG
_predicted", notes = "Training_set_Phase_C_SPN_measured", quant_thr = 0.99
9)
```

In [ ]:
```
#check whether the number of expressed genes has an effect on prediction ac
curacy
ggplot(results_training_phaseC_exp, aes(x = CAG_measured, y = CAG_predicte
d, color = num_expr_genes)) +
geom_point() +
xlim(c(0, 800)) + ylim(c(0, 800)) +
geom_abline()
```

Predict_on_test_set

In [ ]:
```
#create dataframes with model predictions
results_test <- data.frame(CAG_measured = test_data[, 1], CAG_predicted = a
s.numeric(pred_test), pred_phase = pred_test_phase, num_expr_genes = num_ex
pr_genes_test)
CAG_measured_exp_test <- test_data[, 1]
CAG_measured_exp_test[which(CAG_measured_exp_test < 36)] <- NA

results_test_exp <- data.frame(CAG_measured = CAG_measured_exp_test, CAG_pr
edicted = as.numeric(pred_test), pred_phase = pred_test_phase, num_expr_gen
es = num_expr_genes_test)
results_test_phaseC <- results_test[which(results_test$pred_phase == "C-D-
E"), ]
results_test_phaseC_exp <- results_test_exp[which(results_test$pred_phase =
= "C-D-E"), ]
results_test_phaseC_measured <- results_test[which(results_test$CAG_measure
d > 150), ]

#Evaluate_correlation(results_test, "CAG_measured", "CAG_predicted", notes
= "Test_set_All_SPN", quant_thr = 0.999)
Evaluate_correlation(results_test_phaseC, "CAG_measured", "CAG_predicted",
notes = "Test_set_Phase_C_SPN", quant_thr = 0.999)
Evaluate_correlation(results_test_phaseC_exp, "CAG_measured", "CAG_predicte
d", notes = "Test_set_Phase_C_SPN_expandedOnly", quant_thr = 0.999)
Evaluate_correlation(results_test_phaseC_measured, "CAG_measured", "CAG_pre
dicted", notes = "Test_set_Phase_C_SPN_measured", quant_thr = 0.999)
```

In [ ]:
```
#check whether the number of expressed genes has an effect on prediction ac
curacy
ggplot(results_test_phaseC_exp, aes(x = CAG_measured, y = CAG_predicted, co
lor = num_expr_genes)) +
geom_point() +
xlim(c(0, 800)) + ylim(c(0, 800)) +
geom_abline()
```

In [ ]:
```
#QC
#summary(apply(phase_test, 1, function(x) length(which(x > 0))))

table(pred_training_phase)
table(pred_training_filtered_phase)

table(pred_test_phase)

table(predicted_phase_all_HD)
table(predicted_phase_all_HD_noNA)

table(predicted_phase_all_CTRL)
```

Import_Lee_dataset

In [ ]:
```r
#import Lee counts
Lee_counts <- Read10X("PROJECT_ELongATE/Lee/renamed_data")
#create single-cell experiment
Lee_sce <- SingleCellExperiment(list(counts = Lee_counts))
#create Seurat object
Lee <- CreateSeuratObject(counts(Lee_sce), project = "Lee", min.cells = 0,
min.features = 0)
#read metadata
Lee_metadata <- read.table("PROJECT_ELongATE/Lee/renamed_data/GSE152058_hum
an_snRNA_processed_coldata.tsv", header = TRUE, sep = "\t")
#Lee_metadata
#table(Lee_metadata$CellType)
rownames(Lee_metadata) <- Lee_metadata$Barcode
Lee@meta.data <- cbind(Lee@meta.data, Lee_metadata[rownames(Lee@meta.data),
])
#head(Lee@meta.data)
Lee[["percent.mt"]] <- PercentageFeatureSet(Lee, pattern = "^MT-")
```

In [ ]:
```r
#find the number of cells for each MSN subtype
tmp <- lapply(split(Lee_metadata, Lee_metadata$NBB_ID), function(x) {
    x_caudate <- x[which(x$Region == "Caudate"), ]
    x_putamen <- x[which(x$Region == "Putamen"), ]
    num_cells_caudate <- data.frame(num_cells_caudate = dim(x_caudate)[1])
    num_D1_spns_caudate <- data.frame(num_D1_SPN_caudate = dim(x[grep(x = x
_caudate$CellType, pattern = "D1_MSN"), ])[1])
    num_D2_spns_caudate <- data.frame(num_D2_SPN_caudate = dim(x[grep(x = x
_caudate$CellType, pattern = "D2_MSN"), ])[1])
    num_cells_putamen <- data.frame(num_cells_putamen = dim(x_putamen)[1])
    num_D1_spns_putamen <- data.frame(num_D1_SPN_putamen = dim(x[grep(x = x
_putamen$CellType, pattern = "D1_MSN"), ])[1])
    num_D2_spns_putamen <- data.frame(num_D2_SPN_putamen = dim(x[grep(x = x
_putamen$CellType, pattern = "D1_MSN"), ])[1])
    return(data.frame(num_cells_caudate, num_D1_spns_caudate, num_D2_spns_c
audate, num_cells_putamen, num_D1_spns_putamen, num_D2_spns_putamen))
})

num_cells_Lee <- do.call(rbind, tmp)
num_cells_Lee
```

In [ ]:
```r
#subset HD donors
Lee_HD <- subset(x = Lee, subset = Grade != "Control")
Lee_HD_MSN <- subset(x = Lee_HD, subset = CellType == "D1_MSN" | CellType =
= "D2_MSN")
Lee_HD_MSN_Caudate <- subset(x = Lee_HD_MSN, subset = Region == "Caudate")
Lee_HD_MSN_Putamen <- subset(x = Lee_HD_MSN, subset = Region == "Putamen")
head(Lee_HD_MSN@meta.data)
```

In [ ]:
```r
#subset CTRL donors
Lee_CTRL <- subset(x = Lee, subset = Grade == "Control")
Lee_CTRL_MSN <- subset(x = Lee_CTRL, subset = CellType == "D1_MSN" | CellTy
pe == "D2_MSN")
Lee_CTRL_MSN_Caudate <- subset(x = Lee_CTRL_MSN, subset = Region == "Caudat
e")
Lee_CTRL_MSN_Putamen <- subset(x = Lee_CTRL_MSN, subset = Region == "Putame
n")
head(Lee_CTRL_MSN@meta.data)
```

Lee_Caudate_only

In [ ]:
```
#Run SCTransform for HD donors
Lee_HD_MSN_Caudate <- SCTransform(Lee_HD_MSN_Caudate, vars.to.regress = c
("nCount_RNA", "nFeature_RNA", "percent.mt"))
#Lee_MSN_Caudate <- SCTransform(Lee_MSN_Caudate)
cat(sprintf("Num. features = %d; Num. SPN = %d\n", dim(Lee_HD_MSN_Caudate)
[1], dim(Lee_HD_MSN_Caudate)[2]))
#Run PCA and UMAP for HD donors
Lee_HD_MSN_Caudate <- RunPCA(Lee_HD_MSN_Caudate, features = VariableFeature
s(object = Lee_HD_MSN_Caudate), npcs = 50, reduction.name = "pca")
ElbowPlot(Lee_HD_MSN_Caudate, reduction = "pca", ndims = 50)
num_dims <- 40
Lee_HD_MSN_Caudate <- RunUMAP(Lee_HD_MSN_Caudate, dims = 1:num_dims, reduct
ion = "pca", reduction.name = "umap", reduction.key = "umap")
DimPlot(Lee_HD_MSN_Caudate, group.by = "Batch", reduction = "umap", pt.size
= 1)
Lee_HD_MSN_Caudate@meta.data$Batch <- factor(Lee_HD_MSN_Caudate@meta.data$B
atch)
Lee_HD_MSN_Caudate <- RunHarmony(object = Lee_HD_MSN_Caudate, reduction.use
= "pca", group.by.vars = "Batch",
                                 reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Lee_HD_MSN_Caudate <- RunUMAP(Lee_HD_MSN_Caudate, dims = 1:num_dims, reduct
ion = "harmonyPca", reduction.name = "harmony")
DimPlot(Lee_HD_MSN_Caudate, group.by = "Batch", reduction = "harmony", pt.s
ize = 1)
```

In [ ]:
```
#Run SCTransform for CTRL donors
Lee_CTRL_MSN_Caudate <- SCTransform(Lee_CTRL_MSN_Caudate, vars.to.regress =
c("nCount_RNA", "nFeature_RNA", "percent.mt"))
#Lee_CTRL_MSN_Caudate <- SCTransform(Lee_CTRL_MSN_Caudate)
cat(sprintf("Num. features = %d; Num. SPN = %d\n", dim(Lee_CTRL_MSN_Caudat
e)[1], dim(Lee_CTRL_MSN_Caudate)[2]))
#Run PCA and UMAP for CTRL donors
Lee_CTRL_MSN_Caudate <- RunPCA(Lee_CTRL_MSN_Caudate, features = VariableFea
tures(object = Lee_CTRL_MSN_Caudate), npcs = 50, reduction.name = "pca")
ElbowPlot(Lee_CTRL_MSN_Caudate, reduction = "pca", ndims = 50)
num_dims <- 40
Lee_CTRL_MSN_Caudate <- RunUMAP(Lee_CTRL_MSN_Caudate, dims = 1:num_dims, re
duction = "pca", reduction.name = "umap", reduction.key = "umap")
DimPlot(Lee_CTRL_MSN_Caudate, group.by = "Batch", reduction = "umap", pt.si
ze = 1)
Lee_CTRL_MSN_Caudate@meta.data$Batch <- factor(Lee_CTRL_MSN_Caudate@meta.da
ta$Batch)
Lee_CTRL_MSN_Caudate <- RunHarmony(object = Lee_CTRL_MSN_Caudate, reductio
n.use = "pca", group.by.vars = "Batch",
                                 reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Lee_CTRL_MSN_Caudate <- RunUMAP(Lee_CTRL_MSN_Caudate, dims = 1:num_dims, re
duction = "harmonyPca", reduction.name = "harmony")
DimPlot(Lee_CTRL_MSN_Caudate, group.by = "Batch", reduction = "harmony", p
t.size = 1)
```

In [ ]:
```
# saveRDS(object = Lee_HD_MSN_Caudate, file = "PROJECT_ELongATE/Lee/renamed
_data/Lee_HD_MSN_Caudate.rds")
# saveRDS(object = Lee_CTRL_MSN_Caudate, file = "PROJECT_ELongATE/Lee/renam
ed_data/Lee_CTRL_MSN_Caudate.rds")
```

In [ ]:
```
# Lee_HD_MSN_Caudate <- readRDS("PROJECT_ELongATE/Lee/renamed_data/Lee_HD_M
SN_Caudate.rds")
# Lee_CTRL_MSN_Caudate <- readRDS("PROJECT_ELongATE/Lee/renamed_data/Lee_CT
RL_MSN_Caudate.rds")
```

In [ ]:
```
#plots
options(repr.plot.width=20, repr.plot.height=20)
FeaturePlot(Lee_HD_MSN_Caudate, reduction = "harmony", features = "nCount_R
NA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_NCOUNTS.pdf", height = 8, width = 8)
FeaturePlot(Lee_HD_MSN_Caudate, reduction = "harmony", features = "nFeature
_RNA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_NFEATURES.pdf", height = 8, width = 8)
FeaturePlot(Lee_HD_MSN_Caudate, reduction = "harmony", features = "percent.
mt", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_PERCMT.pdf", height = 8, width = 8)
DimPlot(Lee_HD_MSN_Caudate, group.by = "CellType", reduction = "harmony", p
t.size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_CELLTYPE.pdf", height = 8, width = 8)
DimPlot(Lee_HD_MSN_Caudate, group.by = "Batch", reduction = "harmony", pt.s
ize = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_BATCH.pdf", width = 8, height = 8)
DimPlot(Lee_HD_MSN_Caudate, group.by = "NBB_ID", reduction = "harmony", pt.
size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_NBBID.pdf", width = 8, height = 8)
DimPlot(Lee_HD_MSN_Caudate, group.by = "Region", reduction = "harmony", pt.
size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_REGION.pdf", width = 8, height = 8)
```

In [ ]:
```
#extract phase C genes in the model that are also expressed in the dataset
phaseC_genes_Lee_HD_MSN_Caudate <- intersect(colnames(phase_test), rownames
(Lee_HD_MSN_Caudate))
phaseC_genes_notExpLee_HD_MSN_Caudate_names <- setdiff(colnames(phase_tes
t), rownames(Lee_HD_MSN_Caudate))
#phaseC_genes_notExpLee_HD_MSN_Caudate_names
phaseC_genes_notExpLee_HD_MSN_Caudate <- matrix(data = 0, nrow = length(pha
seC_genes_notExpLee_HD_MSN_Caudate_names), ncol = dim(Lee_HD_MSN_Caudate@as
says$SCT$data)[2])
rownames(phaseC_genes_notExpLee_HD_MSN_Caudate) <- phaseC_genes_notExpLee_H
D_MSN_Caudate_names
colnames(phaseC_genes_notExpLee_HD_MSN_Caudate) <- colnames(Lee_HD_MSN_Caud
ate@assays$SCT$data)
phase_counts_Lee_HD_MSN_Caudate <- Lee_HD_MSN_Caudate@assays$SCT$data[phase
C_genes_Lee_HD_MSN_Caudate, ]
phase_counts_Lee_HD_MSN_Caudate <- rbind(phase_counts_Lee_HD_MSN_Caudate, p
haseC_genes_notExpLee_HD_MSN_Caudate)

phaseC_genes_Lee_CTRL_MSN_Caudate <- intersect(colnames(phase_test), rownam
es(Lee_CTRL_MSN_Caudate))
phaseC_genes_notExpLee_CTRL_MSN_Caudate_names <- setdiff(colnames(phase_tes
t), rownames(Lee_CTRL_MSN_Caudate))
#phaseC_genes_notExpLee_CTRL_MSN_Caudate_names
phaseC_genes_notExpLee_CTRL_MSN_Caudate <- matrix(data = 0, nrow = length(p
haseC_genes_notExpLee_CTRL_MSN_Caudate_names), ncol = dim(Lee_CTRL_MSN_Caud
ate@assays$SCT$data)[2])
rownames(phaseC_genes_notExpLee_CTRL_MSN_Caudate) <- phaseC_genes_notExpLee
_CTRL_MSN_Caudate_names
colnames(phaseC_genes_notExpLee_CTRL_MSN_Caudate) <- colnames(Lee_CTRL_MSN_
Caudate@assays$SCT$data)
phase_counts_Lee_CTRL_MSN_Caudate <- Lee_CTRL_MSN_Caudate@assays$SCT$data[p
haseC_genes_Lee_CTRL_MSN_Caudate, ]
phase_counts_Lee_CTRL_MSN_Caudate <- rbind(phase_counts_Lee_CTRL_MSN_Caudat
e, phaseC_genes_notExpLee_CTRL_MSN_Caudate)
```

In [ ]:
```
phase_counts_Lee_HD_MSN_Caudate <- phase_counts_Lee_HD_MSN_Caudate[colnames
(phase_test), ]
phase_test_Lee_HD_MSN_Caudate <- data.frame(t(phase_counts_Lee_HD_MSN_Cauda
te))
#head(phase_test_Lee_HD_MSN_Caudate)
phase_counts_Lee_CTRL_MSN_Caudate <- phase_counts_Lee_CTRL_MSN_Caudate[coln
ames(phase_test), ]
phase_test_Lee_CTRL_MSN_Caudate <- data.frame(t(phase_counts_Lee_CTRL_MSN_C
audate))
#head(phase_test_Lee_CTRL_MSN_Caudate)
```

In [ ]:
```r
#predict phase
class_thr <- 0.5
predicted_phase_test_Lee_HD_MSN_Caudate_wprob <- predict(model_phase, newx = as.matrix(phase_test_Lee_HD_MSN_Caudate), s = lambda_val, type = "response") #glmnet
predicted_phase_test_Lee_HD_MSN_Caudate <- rep("A-B", dim(predicted_phase_test_Lee_HD_MSN_Caudate_wprob)[1])
predicted_phase_test_Lee_HD_MSN_Caudate[which(predicted_phase_test_Lee_HD_MSN_Caudate_wprob > class_thr)] <- "C-D-E" #glmnet
Lee_HD_MSN_Caudate[["PREDICTED_PHASE"]] <- NA
Lee_HD_MSN_Caudate@meta.data[rownames(predicted_phase_test_Lee_HD_MSN_Caudate_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Lee_HD_MSN_Caudate
Lee_HD_MSN_Caudate[["PROB_PHASE_CDE"]] <- NA
Lee_HD_MSN_Caudate@meta.data[rownames(predicted_phase_test_Lee_HD_MSN_Caudate_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Lee_HD_MSN_Caudate_wprob

predicted_phase_test_Lee_CTRL_MSN_Caudate_wprob <- predict(model_phase, newx = as.matrix(phase_test_Lee_CTRL_MSN_Caudate), s = lambda_val, type = "response") #glmnet
predicted_phase_test_Lee_CTRL_MSN_Caudate <- rep("A-B", dim(predicted_phase_test_Lee_CTRL_MSN_Caudate_wprob)[1])
predicted_phase_test_Lee_CTRL_MSN_Caudate[which(predicted_phase_test_Lee_CTRL_MSN_Caudate_wprob > class_thr)] <- "C-D-E" #glmnet
Lee_CTRL_MSN_Caudate[["PREDICTED_PHASE"]] <- NA
Lee_CTRL_MSN_Caudate@meta.data[rownames(predicted_phase_test_Lee_CTRL_MSN_Caudate_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Lee_CTRL_MSN_Caudate
Lee_CTRL_MSN_Caudate[["PROB_PHASE_CDE"]] <- NA
Lee_CTRL_MSN_Caudate@meta.data[rownames(predicted_phase_test_Lee_CTRL_MSN_Caudate_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Lee_CTRL_MSN_Caudate_wprob
```

In [ ]:
```r
#summary(apply(phase_test_Lee_HD_MSN_Caudate, 1, function(x) length(which(x > 0))))
table(predicted_phase_test_Lee_HD_MSN_Caudate)
table(predicted_phase_test_Lee_CTRL_MSN_Caudate)
```

In [ ]:
```r
summary(predicted_phase_test_Lee_HD_MSN_Caudate_wprob)
summary(predicted_phase_test_Lee_CTRL_MSN_Caudate_wprob)
```

In [ ]:
```r
#do plots
DimPlot(Lee_HD_MSN_Caudate, group.by = "PREDICTED_PHASE", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_PREDPHASE.pdf", width = 8, height = 8)
FeaturePlot(Lee_HD_MSN_Caudate, features = "PROB_PHASE_CDE", cols = c("blue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Lee_HD_MSN_Caudate_PROBPREDPHASE.pdf", width = 8, height = 8)
Lee_HD_MSN_Caudate@meta.data$Grade <- factor(paste0("HD", Lee_HD_MSN_Caudate@meta.data$Grade), levels = c("HD4", "HD3", "HD2"))
DimPlot(Lee_HD_MSN_Caudate, group.by = "Grade", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Caudate_GRADE.pdf", width = 8, height = 8)
```

In [ ]:
```
#do plots
DimPlot(Lee_CTRL_MSN_Caudate, group.by = "PREDICTED_PHASE", reduction = "ha
rmony", pt.size = 1)
ggsave("UMAP_Lee_CTRL_MSN_Caudate_PREDPHASE.pdf", width = 8, height = 8)
FeaturePlot(Lee_CTRL_MSN_Caudate, features = "PROB_PHASE_CDE", cols = c("bl
ue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Lee_CTRL_MSN_Caudate_PROBPREDPHASE.pdf", width = 8, height =
8)
DimPlot(Lee_CTRL_MSN_Caudate, group.by = "Batch", reduction = "harmony", p
t.size = 1)
ggsave("UMAP_Lee_CTRL_MSN_Caudate_BATCH.pdf", width = 8, height = 8)
```

In [ ]:
```
#create metadata file for plotting
Lee_Caudate_Grade_CDE_tmp <- lapply(split(Lee_HD_MSN_Caudate@meta.data, Lee
_HD_MSN_Caudate@meta.data$NBB_ID), function(x) {
    GRADE <- factor(unique(x$Grade), levels = c("HD1", "HD2", "HD3", "HD
4"))
    SN <- unique(x$NBB_ID)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    tmp <- lapply(split(x, x$CellType), function(y) {
        NUM_MSN_CT <- length(y$PREDICTED_PHASE)
        NUM_MSN_CT_SQ <- NUM_MSN_CT**2
        CELL_TYPE <- y$CellType
        if (length(unique(y$PREDICTED_PHASE)) > 1) {
          val <- 100*table(y$PREDICTED_PHASE)[2]/(table(y$PREDICTED_PHASE)
[1] + table(y$PREDICTED_PHASE)[2])
        } else {
          val <- 0
        }
        names(val) <- "Fract_CDE";
        Fract_CDE <- rep(val, NUM_MSN_CT_SQ);
    return(data.frame(NUM_MSN_CT_SQ, CELL_TYPE, Fract_CDE))})
    df <- do.call(rbind, tmp)
    #print(df)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    #Fract_CDE <- 100*length(which(x$PREDICTED_PHASE == "C-D-E"))/length(x
$PREDICTED_PHASE)
    #CELL_TYPE <- x$sub_type_4
    #return(df)
    #return(data.frame(SAMPLE = rep(SN, df$NUM_MSN_CT_SQ), GRADE = rep(GRAD
E, df$NUM_MSN_CT_SQ), CELL_TYPE = df$CELL_TYPE, Fract_CDE = df$Fract_CDE, N
UM_MSN_SQ=df$NUM_MSN_CT_SQ))
    return(data.frame(SAMPLE = SN, GRADE = GRADE, CELL_TYPE = df$CELL_TYPE,
Fract_CDE = df$Fract_CDE, NUM_MSN_SQ=df$NUM_MSN_CT_SQ))
})
Lee_Caudate_Grade_CDE <- do.call(rbind, Lee_Caudate_Grade_CDE_tmp)
Lee_Caudate_Grade_CDE <- Lee_Caudate_Grade_CDE[order(Lee_Caudate_Grade_CDE
$GRADE), ]
Lee_Caudate_Grade_CDE$SAMPLE <- factor(Lee_Caudate_Grade_CDE$SAMPLE, levels
= unique(Lee_Caudate_Grade_CDE$SAMPLE))
#head(Lee_Caudate_Grade_CDE)
uni_Lee_Caudate_Grade_CDE <- unique(Lee_Caudate_Grade_CDE)
uni_Lee_Caudate_Grade_CDE$NUM_SPN <- sqrt(uni_Lee_Caudate_Grade_CDE$NUM_MSN
_SQ)
#uni_Lee_Caudate_Grade_CDE
```

In [ ]:
```
#num_cells_Lee
Fract_SPN_Caudate <- c(num_cells_Lee["3730", "num_D1_SPN_caudate"]/num_cell
s_Lee["3730", "num_cells_caudate"],
                       num_cells_Lee["3730", "num_D2_SPN_caudate"]/num_cell
s_Lee["3730", "num_cells_caudate"],
                       num_cells_Lee["3881", "num_D1_SPN_caudate"]/num_cell
s_Lee["3881", "num_cells_caudate"],
                       num_cells_Lee["3881", "num_D2_SPN_caudate"]/num_cell
s_Lee["3881", "num_cells_caudate"],
                       num_cells_Lee["2030", "num_D1_SPN_caudate"]/num_cell
s_Lee["2030", "num_cells_caudate"],
                       num_cells_Lee["2030", "num_D2_SPN_caudate"]/num_cell
s_Lee["2030", "num_cells_caudate"],
                       num_cells_Lee["2952", "num_D1_SPN_caudate"]/num_cell
s_Lee["2952", "num_cells_caudate"],
                       num_cells_Lee["2952", "num_D2_SPN_caudate"]/num_cell
s_Lee["2952", "num_cells_caudate"],
                       num_cells_Lee["4254", "num_D1_SPN_caudate"]/num_cell
s_Lee["4254", "num_cells_caudate"],
                       num_cells_Lee["4254", "num_D2_SPN_caudate"]/num_cell
s_Lee["4254", "num_cells_caudate"],
                       num_cells_Lee["2665", "num_D1_SPN_caudate"]/num_cell
s_Lee["2665", "num_cells_caudate"],
                       num_cells_Lee["2665", "num_D2_SPN_caudate"]/num_cell
s_Lee["2665", "num_cells_caudate"])
uni_Lee_Caudate_Grade_CDE$Fract_SPN <- Fract_SPN_Caudate
uni_Lee_Caudate_Grade_CDE
```

In [ ]:
```
#plot fraction of SPN in C-D-E phase by HD grade
ggplot(uni_Lee_Caudate_Grade_CDE, aes(x = GRADE, y = Fract_CDE, color = CEL
L_TYPE)) +
geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(w
idth = 0.1, height = 0)) +
scale_color_manual(values = c("#F8766D", "#619CFF")) +
theme_classic() +
xlab("HD GRADE") +
ylim(c(0, 100)) +
ylab("Fraction cells in C-D-E phase (%)")
ggsave("Lee_Caudate_GRADE_vs_fraction_SPN_CDE.pdf", width = 6, height = 6)
```

In [ ]:
```
# ggplot(uni_Lee_Caudate_Grade_CDE[which(uni_Lee_Caudate_Grade_CDE$NUM_SPN
> 50), ], aes(x = GRADE, y = Fract_CDE, color = CELL_TYPE)) +
# geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter
(width = 0.1, height = 0)) +
# scale_color_manual(values = c("#F8766D", "#619CFF")) +
# theme_classic() +
# xlab("HD GRADE") +
# ylim(c(0, 100)) +
# ylab("Fraction cells in C-D-E phase (%)")
# ggsave("Lee_Caudate_GRADE_vs_fraction_SPN_CDE_gt50.pdf", width = 6, heigh
t = 6)
```

Lee_Putamen_only

In [ ]:
```
#Run SCTransform for HD donors
Lee_HD_MSN_Putamen <- SCTransform(Lee_HD_MSN_Putamen, vars.to.regress = c
("nCount_RNA", "nFeature_RNA", "percent.mt"))
#Lee_HD_MSN_Putamen <- SCTransform(Lee_HD_MSN_Putamen)
cat(sprintf("Num. features = %d; Num. SPN = %d\n", dim(Lee_HD_MSN_Putamen)
[1], dim(Lee_HD_MSN_Putamen)[2]))
#Run PCA and UMAP for HD donors
Lee_HD_MSN_Putamen <- RunPCA(Lee_HD_MSN_Putamen, features = VariableFeature
s(object = Lee_HD_MSN_Putamen), npcs = 50, reduction.name = "pca")
ElbowPlot(Lee_HD_MSN_Putamen, reduction = "pca", ndims = 50)
num_dims <- 40
Lee_HD_MSN_Putamen <- RunUMAP(Lee_HD_MSN_Putamen, dims = 1:num_dims, reduct
ion = "pca", reduction.name = "umap", reduction.key = "umap")
DimPlot(Lee_HD_MSN_Putamen, group.by = "Batch", reduction = "umap", pt.size
= 1)
Lee_HD_MSN_Putamen@meta.data$Batch <- factor(Lee_HD_MSN_Putamen@meta.data$B
atch)
Lee_HD_MSN_Putamen <- RunHarmony(object = Lee_HD_MSN_Putamen, reduction.use
= "pca", group.by.vars = "Batch",
                                 reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Lee_HD_MSN_Putamen <- RunUMAP(Lee_HD_MSN_Putamen, dims = 1:num_dims, reduct
ion = "harmonyPca", reduction.name = "harmony")
DimPlot(Lee_HD_MSN_Putamen, group.by = "Batch", reduction = "harmony", pt.s
ize = 1)
```

In [ ]:
```
#Run SCTransform for CTRL donors
Lee_CTRL_MSN_Putamen <- SCTransform(Lee_CTRL_MSN_Putamen, vars.to.regress =
c("nCount_RNA", "nFeature_RNA", "percent.mt"))
#Lee_CTRL_MSN_Putamen <- SCTransform(Lee_CTRL_MSN_Putamen)
cat(sprintf("Num. features = %d; Num. SPN = %d\n", dim(Lee_CTRL_MSN_Putame
n)[1], dim(Lee_CTRL_MSN_Putamen)[2]))
#Run PCA and UMAP for CTRL donors
Lee_CTRL_MSN_Putamen <- RunPCA(Lee_CTRL_MSN_Putamen, features = VariableFea
tures(object = Lee_CTRL_MSN_Putamen), npcs = 50, reduction.name = "pca")
ElbowPlot(Lee_CTRL_MSN_Putamen, reduction = "pca", ndims = 50)
num_dims <- 40
Lee_CTRL_MSN_Putamen <- RunUMAP(Lee_CTRL_MSN_Putamen, dims = 1:num_dims, re
duction = "pca", reduction.name = "umap", reduction.key = "umap")
DimPlot(Lee_CTRL_MSN_Putamen, group.by = "Batch", reduction = "umap", pt.si
ze = 1)
Lee_CTRL_MSN_Putamen@meta.data$Batch <- factor(Lee_CTRL_MSN_Putamen@meta.da
ta$Batch)
Lee_CTRL_MSN_Putamen <- RunHarmony(object = Lee_CTRL_MSN_Putamen, reductio
n.use = "pca", group.by.vars = "Batch",
                                 reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Lee_CTRL_MSN_Putamen <- RunUMAP(Lee_CTRL_MSN_Putamen, dims = 1:num_dims, re
duction = "harmonyPca", reduction.name = "harmony")
DimPlot(Lee_CTRL_MSN_Putamen, group.by = "Batch", reduction = "harmony", p
t.size = 1)
```

In [ ]:
```
# saveRDS(object = Lee_HD_MSN_Putamen, file = "PROJECT_ELongATE/Lee/renamed
_data/Lee_HD_MSN_Putamen.rds")
# saveRDS(object = Lee_CTRL_MSN_Putamen, file = "PROJECT_ELongATE/Lee/renam
ed_data/Lee_CTRL_MSN_Putamen.rds")
```

In [ ]:
```
# Lee_HD_MSN_Putamen <- readRDS("PROJECT_ELongATE/Lee/renamed_data/Lee_HD_MSN_Putamen.rds")
# Lee_CTRL_MSN_Putamen <- readRDS("PROJECT_ELongATE/Lee/renamed_data/Lee_CTRL_MSN_Putamen.rds")
```

In [ ]:
```
#do plots
options(repr.plot.width=20, repr.plot.height=20)
FeaturePlot(Lee_HD_MSN_Putamen, reduction = "harmony", features = "nCount_RNA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_NCOUNTS.pdf", height = 8, width = 8)
FeaturePlot(Lee_HD_MSN_Putamen, reduction = "harmony", features = "nFeature_RNA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_NFEATURES.pdf", height = 8, width = 8)
FeaturePlot(Lee_HD_MSN_Putamen, reduction = "harmony", features = "percent.mt", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_PERCMT.pdf", height = 8, width = 8)
DimPlot(Lee_HD_MSN_Putamen, group.by = "CellType", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_CELLTYPE.pdf", height = 8, width = 8)
DimPlot(Lee_HD_MSN_Putamen, group.by = "Batch", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_BATCH.pdf", width = 8, height = 8)
DimPlot(Lee_HD_MSN_Putamen, group.by = "NBB_ID", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_NBBID.pdf", width = 8, height = 8)
DimPlot(Lee_HD_MSN_Putamen, group.by = "Region", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_REGION.pdf", width = 8, height = 8)
```

In [ ]:
```
#extract phase C genes in the model that are also expressed in the dataset
phaseC_genes_Lee_HD_MSN_Putamen <- intersect(colnames(phase_test), rownames
(Lee_HD_MSN_Putamen))
phaseC_genes_notExpLee_HD_MSN_Putamen_names <- setdiff(colnames(phase_tes
t), rownames(Lee_HD_MSN_Putamen))
#phaseC_genes_notExpLee_HD_MSN_Putamen_names
phaseC_genes_notExpLee_HD_MSN_Putamen <- matrix(data = 0, nrow = length(pha
seC_genes_notExpLee_HD_MSN_Putamen_names), ncol = dim(Lee_HD_MSN_Putamen@as
says$SCT$data)[2])
rownames(phaseC_genes_notExpLee_HD_MSN_Putamen) <- phaseC_genes_notExpLee_H
D_MSN_Putamen_names
colnames(phaseC_genes_notExpLee_HD_MSN_Putamen) <- colnames(Lee_HD_MSN_Puta
men@assays$SCT$data)
phase_counts_Lee_HD_MSN_Putamen <- Lee_HD_MSN_Putamen@assays$SCT$data[phase
C_genes_Lee_HD_MSN_Putamen, ]
phase_counts_Lee_HD_MSN_Putamen <- rbind(phase_counts_Lee_HD_MSN_Putamen, p
haseC_genes_notExpLee_HD_MSN_Putamen)

phaseC_genes_Lee_CTRL_MSN_Putamen <- intersect(colnames(phase_test), rownam
es(Lee_CTRL_MSN_Putamen))
phaseC_genes_notExpLee_CTRL_MSN_Putamen_names <- setdiff(colnames(phase_tes
t), rownames(Lee_CTRL_MSN_Putamen))
#phaseC_genes_notExpLee_CTRL_MSN_Putamen_names
phaseC_genes_notExpLee_CTRL_MSN_Putamen <- matrix(data = 0, nrow = length(p
haseC_genes_notExpLee_CTRL_MSN_Putamen_names), ncol = dim(Lee_CTRL_MSN_Puta
men@assays$SCT$data)[2])
rownames(phaseC_genes_notExpLee_CTRL_MSN_Putamen) <- phaseC_genes_notExpLee
_CTRL_MSN_Putamen_names
colnames(phaseC_genes_notExpLee_CTRL_MSN_Putamen) <- colnames(Lee_CTRL_MSN_
Putamen@assays$SCT$data)

phase_counts_Lee_CTRL_MSN_Putamen <- Lee_CTRL_MSN_Putamen@assays$SCT$data[p
haseC_genes_Lee_CTRL_MSN_Putamen, ]
phase_counts_Lee_CTRL_MSN_Putamen <- rbind(phase_counts_Lee_CTRL_MSN_Putame
n, phaseC_genes_notExpLee_CTRL_MSN_Putamen)
```

In [ ]:
```
phase_counts_Lee_HD_MSN_Putamen <- phase_counts_Lee_HD_MSN_Putamen[colnames
(phase_test), ]
phase_test_Lee_HD_MSN_Putamen <- data.frame(t(phase_counts_Lee_HD_MSN_Putam
en))
#head(phase_test_Lee_HD_MSN_Putamen)
phase_counts_Lee_CTRL_MSN_Putamen <- phase_counts_Lee_CTRL_MSN_Putamen[coln
ames(phase_test), ]
phase_test_Lee_CTRL_MSN_Putamen <- data.frame(t(phase_counts_Lee_CTRL_MSN_P
utamen))
#head(phase_test_Lee_CTRL_MSN_Putamen)
```

In [ ]:
```r
#predict phase
class_thr <- 0.5
predicted_phase_test_Lee_HD_MSN_Putamen_wprob <- predict(model_phase, newx = as.matrix(phase_test_Lee_HD_MSN_Putamen), s = lambda_val, type = "response") #glmnet
predicted_phase_test_Lee_HD_MSN_Putamen <- rep("A-B", dim(predicted_phase_test_Lee_HD_MSN_Putamen_wprob)[1])
predicted_phase_test_Lee_HD_MSN_Putamen[which(predicted_phase_test_Lee_HD_MSN_Putamen_wprob > class_thr)] <- "C-D-E" #glmnet
Lee_HD_MSN_Putamen[["PREDICTED_PHASE"]] <- NA
Lee_HD_MSN_Putamen@meta.data[rownames(predicted_phase_test_Lee_HD_MSN_Putamen_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Lee_HD_MSN_Putamen
Lee_HD_MSN_Putamen[["PROB_PHASE_CDE"]] <- NA
Lee_HD_MSN_Putamen@meta.data[rownames(predicted_phase_test_Lee_HD_MSN_Putamen_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Lee_HD_MSN_Putamen_wprob

predicted_phase_test_Lee_CTRL_MSN_Putamen_wprob <- predict(model_phase, newx = as.matrix(phase_test_Lee_CTRL_MSN_Putamen), s = lambda_val, type = "response") #glmnet
predicted_phase_test_Lee_CTRL_MSN_Putamen <- rep("A-B", dim(predicted_phase_test_Lee_CTRL_MSN_Putamen_wprob)[1])
predicted_phase_test_Lee_CTRL_MSN_Putamen[which(predicted_phase_test_Lee_CTRL_MSN_Putamen_wprob > class_thr)] <- "C-D-E" #glmnet
Lee_CTRL_MSN_Putamen[["PREDICTED_PHASE"]] <- NA
Lee_CTRL_MSN_Putamen@meta.data[rownames(predicted_phase_test_Lee_CTRL_MSN_Putamen_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Lee_CTRL_MSN_Putamen
Lee_CTRL_MSN_Putamen[["PROB_PHASE_CDE"]] <- NA
Lee_CTRL_MSN_Putamen@meta.data[rownames(predicted_phase_test_Lee_CTRL_MSN_Putamen_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Lee_CTRL_MSN_Putamen_wprob
```

In [ ]:
```r
#summary(apply(phase_test_Lee_HD_MSN_Putamen, 1, function(x) length(which(x > 0))))
table(predicted_phase_test_Lee_HD_MSN_Putamen)
table(predicted_phase_test_Lee_CTRL_MSN_Putamen)
```

In [ ]:
```r
summary(predicted_phase_test_Lee_HD_MSN_Putamen_wprob)
summary(predicted_phase_test_Lee_CTRL_MSN_Putamen_wprob)
```

In [ ]:
```r
#do plots
DimPlot(Lee_HD_MSN_Putamen, group.by = "PREDICTED_PHASE", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_PREDPHASE.pdf", width = 8, height = 8)
FeaturePlot(Lee_HD_MSN_Putamen, features = "PROB_PHASE_CDE", cols = c("blue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Lee_HD_MSN_Putamen_PROBPREDPHASE.pdf", width = 8, height = 8)
Lee_HD_MSN_Putamen@meta.data$Grade <- factor(paste0("HD", Lee_HD_MSN_Putamen@meta.data$Grade), levels = c("HD4", "HD3", "HD2"))
DimPlot(Lee_HD_MSN_Putamen, group.by = "Grade", reduction = "harmony", pt.size = 1)
ggsave("UMAP_Lee_HD_MSN_Putamen_GRADE.pdf", width = 8, height = 8)
```

In [ ]:
```
#do plots
DimPlot(Lee_CTRL_MSN_Putamen, group.by = "PREDICTED_PHASE", reduction = "ha
rmony", pt.size = 1)
ggsave("UMAP_Lee_CTRL_MSN_Putamen_PREDPHASE.pdf", width = 8, height = 8)
FeaturePlot(Lee_CTRL_MSN_Putamen, features = "PROB_PHASE_CDE", cols = c("bl
ue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Lee_CTRL_MSN_Putamen_PROBPREDPHASE.pdf", width = 8, height =
8)
DimPlot(Lee_CTRL_MSN_Putamen, group.by = "Batch", reduction = "harmony", p
t.size = 1)
ggsave("UMAP_Lee_CTRL_MSN_Putamen_BATCH.pdf", width = 8, height = 8)
```

In [ ]:
```
#create metadata file for plotting
Lee_Putamen_Grade_CDE_tmp <- lapply(split(Lee_HD_MSN_Putamen@meta.data, Lee
_HD_MSN_Putamen@meta.data$NBB_ID), function(x) {
    GRADE <- factor(unique(x$Grade), levels = c("HD1", "HD2", "HD3", "HD
4"))
    SN <- unique(x$NBB_ID)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    tmp <- lapply(split(x, x$CellType), function(y) {
        NUM_MSN_CT <- length(y$PREDICTED_PHASE)
        NUM_MSN_CT_SQ <- NUM_MSN_CT**2
        CELL_TYPE <- y$CellType
        if (length(unique(y$PREDICTED_PHASE)) > 1) {
          val <- 100*table(y$PREDICTED_PHASE)[2]/(table(y$PREDICTED_PHASE)
[1] + table(y$PREDICTED_PHASE)[2])
        } else {
          val <- 0
        }
        names(val) <- "Fract_CDE";
        Fract_CDE <- rep(val, NUM_MSN_CT_SQ);
    return(data.frame(NUM_MSN_CT_SQ, CELL_TYPE, Fract_CDE))})
    df <- do.call(rbind, tmp)
    #print(df)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    #Fract_CDE <- 100*length(which(x$PREDICTED_PHASE == "C-D-E"))/length(x
$PREDICTED_PHASE)
    #CELL_TYPE <- x$sub_type_4
    #return(df)
    #return(data.frame(SAMPLE = rep(SN, df$NUM_MSN_CT_SQ), GRADE = rep(GRAD
E, df$NUM_MSN_CT_SQ), CELL_TYPE = df$CELL_TYPE, Fract_CDE = df$Fract_CDE, N
UM_MSN_SQ=df$NUM_MSN_CT_SQ))
    return(data.frame(SAMPLE = SN, GRADE = GRADE, CELL_TYPE = df$CELL_TYPE,
Fract_CDE = df$Fract_CDE, NUM_MSN_SQ=df$NUM_MSN_CT_SQ))
})
Lee_Putamen_Grade_CDE <- do.call(rbind, Lee_Putamen_Grade_CDE_tmp)
Lee_Putamen_Grade_CDE <- Lee_Putamen_Grade_CDE[order(Lee_Putamen_Grade_CDE
$GRADE), ]
Lee_Putamen_Grade_CDE$SAMPLE <- factor(Lee_Putamen_Grade_CDE$SAMPLE, levels
= unique(Lee_Putamen_Grade_CDE$SAMPLE))
#head(Lee_Putamen_Grade_CDE)
uni_Lee_Putamen_Grade_CDE <- unique(Lee_Putamen_Grade_CDE)
uni_Lee_Putamen_Grade_CDE$NUM_SPN <- sqrt(uni_Lee_Putamen_Grade_CDE$NUM_MSN
_SQ)
#uni_Lee_Putamen_Grade_CDE
```

In [ ]:
```
Fract_SPN_Putamen <- c(num_cells_Lee["3881", "num_D1_SPN_putamen"]/num_cell
s_Lee["3881", "num_cells_putamen"],
                       num_cells_Lee["3881", "num_D2_SPN_putamen"]/num_cell
s_Lee["3881", "num_cells_putamen"],
                       num_cells_Lee["4225", "num_D1_SPN_putamen"]/num_cell
s_Lee["4225", "num_cells_putamen"],
                       num_cells_Lee["4225", "num_D2_SPN_putamen"]/num_cell
s_Lee["4225", "num_cells_putamen"],
                       num_cells_Lee["2952", "num_D1_SPN_putamen"]/num_cell
s_Lee["2952", "num_cells_putamen"],
                       num_cells_Lee["2952", "num_D2_SPN_putamen"]/num_cell
s_Lee["2952", "num_cells_putamen"],
                       num_cells_Lee["4254", "num_D1_SPN_putamen"]/num_cell
s_Lee["4254", "num_cells_putamen"],
                       num_cells_Lee["4254", "num_D2_SPN_putamen"]/num_cell
s_Lee["4254", "num_cells_putamen"],
                       num_cells_Lee["2665", "num_D1_SPN_putamen"]/num_cell
s_Lee["2665", "num_cells_putamen"],
                       #num_cells_Lee["2665", "num_D2_SPN_putamen"]/num_cel
ls_Lee["2665", "num_cells_putamen"],
                       num_cells_Lee["2903", "num_D1_SPN_putamen"]/num_cell
s_Lee["2903", "num_cells_putamen"],
                       num_cells_Lee["2903", "num_D2_SPN_putamen"]/num_cell
s_Lee["2665", "num_cells_putamen"])
uni_Lee_Putamen_Grade_CDE$Fract_SPN <- Fract_SPN_Putamen
uni_Lee_Putamen_Grade_CDE
```

In [ ]:
```
#plot fraction of SPN in C-D-E phase by HD grade
ggplot(uni_Lee_Putamen_Grade_CDE, aes(x = GRADE, y = Fract_CDE, color = CEL
L_TYPE)) +
geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(w
idth = 0.1, height = 0)) +
scale_color_manual(values = c("#F8766D", "#619CFF")) +
theme_classic() +
xlab("HD GRADE") +
ylim(c(0, 100)) +
ylab("Fraction cells in C-D-E phase (%)")
ggsave("Lee_Putamen_GRADE_vs_fraction_SPN_CDE.pdf", width = 6, height = 6)
```

In [ ]:
```
# ggplot(uni_Lee_Putamen_Grade_CDE[which(uni_Lee_Putamen_Grade_CDE$NUM_SPN
> 50), ], aes(x = GRADE, y = Fract_CDE, color = CELL_TYPE)) +
# geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter
(width = 0.1, height = 0)) +
# scale_color_manual(values = c("#F8766D", "#619CFF")) +
# theme_classic() +
# xlab("HD GRADE") +
# ylim(c(0, 100)) +
# ylab("Fraction cells in C-D-E phase (%)")
# ggsave("Lee_Putamen_GRADE_vs_fraction_SPN_CDE_gt50.pdf", width = 6, heigh
t = 6)
```

In [ ]:
```
str(Lee_Caudate_Grade_CDE)
head(Lee_Caudate_Grade_CDE)
str(Lee_Putamen_Grade_CDE)
head(Lee_Putamen_Grade_CDE)
```

In [ ]:
```
colnames(Lee_HD_MSN_Caudate@meta.data)
colnames(Lee_CTRL_MSN_Caudate@meta.data)
colnames(Lee_HD_MSN_Putamen@meta.data)
colnames(Lee_CTRL_MSN_Putamen@meta.data)
```

Import_Paryani_dataset

In [ ]:
```
Paryani <- readRDS("PROJECT_ELongATE/Paryani/neuron_hd_obj_acc_caud_paper.rds")
```

In [ ]:
```
head(Paryani@meta.data)
table(Paryani@meta.data$broad_type)
table(Paryani@meta.data$sub_type_4)
table(Paryani@meta.data$Condition)
table(Paryani@meta.data$CAG)
table(Paryani@meta.data$Batch)
```

In [ ]:
```
#add metadata
tmp <- lapply(split(Paryani@meta.data, Paryani@meta.data$Donor), function(x) {
    x_caudate <- x[which(x$Region == "Caudate"), ]
    x_accumbens <- x[which(x$Region == "Accumbens"), ]
    num_cells_caudate <- data.frame(num_cells_caudate = dim(x_caudate)[1])
    num_dspn1_caudate <- data.frame(num_dSPN1_caudate = dim(x[grep(x = x_caudate$sub_type_4, pattern = "dSPN_1"), ])[1])
    num_dspn2_caudate <- data.frame(num_dSPN2_caudate = dim(x[grep(x = x_caudate$sub_type_4, pattern = "dSPN_2"), ])[1])
    num_ispn1_caudate <- data.frame(num_iSPN1_caudate = dim(x[grep(x = x_caudate$sub_type_4, pattern = "iSPN_1"), ])[1])
    num_ispn2_caudate <- data.frame(num_iSPN2_caudate = dim(x[grep(x = x_caudate$sub_type_4, pattern = "iSPN_2"), ])[1])
    num_cells_accumbens <- data.frame(num_cells_accumbens = dim(x_accumbens)[1])
    num_dspn1_accumbens <- data.frame(num_dSPN1_accumbens = dim(x[grep(x = x_accumbens$sub_type_4, pattern = "dSPN_1"), ])[1])
    num_dspn2_accumbens <- data.frame(num_dSPN2_accumbens = dim(x[grep(x = x_accumbens$sub_type_4, pattern = "dSPN_2"), ])[1])
    num_ispn1_accumbens <- data.frame(num_iSPN1_accumbens = dim(x[grep(x = x_accumbens$sub_type_4, pattern = "iSPN_1"), ])[1])
    num_ispn2_accumbens <- data.frame(num_iSPN2_accumbens = dim(x[grep(x = x_accumbens$sub_type_4, pattern = "iSPN_2"), ])[1])
    return(data.frame(num_cells_caudate, num_dspn1_caudate, num_dspn2_caudate, num_ispn1_caudate, num_ispn2_caudate, num_cells_accumbens, num_dspn1_accumbens, num_dspn2_accumbens, num_ispn1_accumbens, num_ispn2_accumbens))
})

num_cells_Paryani <- do.call(rbind, tmp)
#num_cells_Paryani
```

In [ ]:
```
#subset SPN from HD donors
Paryani_HD <- subset(x = Paryani, subset = Condition == "HD")
Paryani_HD_MSN <- subset(x = Paryani_HD, subset = sub_type_4 == "dSPN_1" |
sub_type_4 == "dSPN_2" | sub_type_4 == "iSPN_1" | sub_type_4 == "iSPN_2")
Paryani_HD_MSN_Accumbens <- subset(x = Paryani_HD_MSN, subset = Region ==
"Accumbens")
Paryani_HD_MSN_Caudate <- subset(x = Paryani_HD_MSN, subset = Region == "Ca
udate")
```

In [ ]:
```
#subset SPN from CTRL donors
Paryani_CTRL <- subset(x = Paryani, subset = Condition != "HD")
Paryani_CTRL_MSN <- subset(x = Paryani_CTRL, subset = sub_type_4 == "dSPN_
1" | sub_type_4 == "dSPN_2" | sub_type_4 == "iSPN_1" | sub_type_4 == "iSPN_
2")
Paryani_CTRL_MSN_Accumbens <- subset(x = Paryani_CTRL_MSN, subset = Region
== "Accumbens")
Paryani_CTRL_MSN_Caudate <- subset(x = Paryani_CTRL_MSN, subset = Region ==
"Caudate")
```

In [ ]:
```
Paryani_HD_MSN_Caudate@meta.data
```

Paryani_Caudate_only

In [ ]:
```
#Run SCTransform for HD donors
Paryani_HD_MSN_Caudate <- SCTransform(Paryani_HD_MSN_Caudate, vars.to.regre
ss = c("nCount_RNA", "nFeature_RNA"))
#Run PCA and UMAP for HD donors
Paryani_HD_MSN_Caudate <- RunPCA(Paryani_HD_MSN_Caudate, features = Variabl
eFeatures(object = Paryani_HD_MSN_Caudate), npcs = 50, reduction.name = "pc
a")
ElbowPlot(Paryani_HD_MSN, reduction = "pca", ndims = 50)
num_dims <- 40
Paryani_HD_MSN_Caudate <- RunUMAP(Paryani_HD_MSN_Caudate, dims = 1:num_dim
s, reduction = "pca", reduction.name = "umap", reduction.key = "umap")
DimPlot(Paryani_HD_MSN_Caudate, group.by = "Batch", reduction = "umap", pt.
size = 3)
Paryani_HD_MSN_Caudate <- RunHarmony(object = Paryani_HD_MSN_Caudate, reduc
tion.use = "pca", group.by.vars = "Batch",
                                reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Paryani_HD_MSN_Caudate <- RunUMAP(Paryani_HD_MSN_Caudate, dims = 1:num_dim
s, reduction = "harmonyPca", reduction.name = "harmony", reduction.key = "h
armony")
```

In [ ]:
```
#Run SCTransform for CTRL donors
Paryani_CTRL_MSN_Caudate <- SCTransform(Paryani_CTRL_MSN_Caudate, vars.to.r
egress = c("nCount_RNA", "nFeature_RNA"))
#Run PCA and UMAP for CTRL donors
Paryani_CTRL_MSN_Caudate <- RunPCA(Paryani_CTRL_MSN_Caudate, features = Var
iableFeatures(object = Paryani_CTRL_MSN_Caudate), npcs = 50, reduction.name
= "pca")
ElbowPlot(Paryani_CTRL_MSN, reduction = "pca", ndims = 50)
num_dims <- 40
Paryani_CTRL_MSN_Caudate <- RunUMAP(Paryani_CTRL_MSN_Caudate, dims = 1:num_
dims, reduction = "pca", reduction.name = "umap", reduction.key = "umap")
DimPlot(Paryani_CTRL_MSN_Caudate, group.by = "Batch", reduction = "umap", p
t.size = 3)
Paryani_CTRL_MSN_Caudate <- RunHarmony(object = Paryani_CTRL_MSN_Caudate, r
eduction.use = "pca", group.by.vars = "Batch",
                              reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Paryani_CTRL_MSN_Caudate <- RunUMAP(Paryani_CTRL_MSN_Caudate, dims = 1:num_
dims, reduction = "harmonyPca", reduction.name = "harmony", reduction.key =
"harmony")
```

In [ ]:
```
#do plots
FeaturePlot(Paryani_HD_MSN_Caudate, reduction = "harmony", features = "nCou
nt_RNA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_NCOUNTS.pdf", width = 8, height = 8)
FeaturePlot(Paryani_HD_MSN_Caudate, reduction = "harmony", features = "nFea
ture_RNA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_NFEATURES.pdf", width = 8, height = 8)
Paryani_HD_MSN_Caudate@meta.data$GERM_EXPCAGLENGTH <- as.numeric(gsub(x = P
aryani_HD_MSN_Caudate@meta.data$CAG, pattern = "-.*", replacement = ""))
FeaturePlot(Paryani_HD_MSN_Caudate, reduction = "harmony", features = "GERM
_EXPCAGLENGTH", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_GERM_EXPCAGLENGTH.pdf", width = 8, heig
ht = 8)
DimPlot(Paryani_HD_MSN_Caudate, group.by = "sub_type_4", reduction = "harmo
ny", pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_CELLTYPE.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Caudate, group.by = "CAG", reduction = "harmony", p
t.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_CAG.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Caudate, group.by = "Batch", reduction = "harmony",
pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_BATCH.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Caudate, group.by = "Region", reduction = "harmony",
pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_REGION.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Caudate, group.by = "Donor", reduction = "harmony",
pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_DONOR.pdf", width = 8, height = 8)
Paryani_HD_MSN_Caudate@meta.data$Grade <- factor(Paryani_HD_MSN_Caudate@met
a.data$Grade, levels = c("HDJ", "HD4", "HD3", "HD2", "HD1", "Control"))
DimPlot(Paryani_HD_MSN_Caudate, group.by = "Grade", reduction = "harmony",
pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_GRADE.pdf", width = 8, height = 8)
```

In [ ]:
```
#extract phase C genes in the model that are also expressed in the dataset
phaseC_genes_Paryani_HD_MSN_Caudate <- intersect(colnames(phase_test), rownames(Paryani_HD_MSN_Caudate))
phaseC_genes_notExpParyani_HD_MSN_Caudate_names <- setdiff(colnames(phase_test), rownames(Paryani_HD_MSN_Caudate))
phaseC_genes_notExpParyani_HD_MSN_Caudate <- matrix(data = 0, nrow = length(phaseC_genes_notExpParyani_HD_MSN_Caudate_names), ncol = dim(Paryani_HD_MSN_Caudate@assays$SCT$data)[2])
rownames(phaseC_genes_notExpParyani_HD_MSN_Caudate) <- phaseC_genes_notExpParyani_HD_MSN_Caudate_names
colnames(phaseC_genes_notExpParyani_HD_MSN_Caudate) <- colnames(Paryani_HD_MSN_Caudate@assays$SCT$data)
phase_counts_Paryani_HD_MSN_Caudate <- Paryani_HD_MSN_Caudate@assays$SCT$data[phaseC_genes_Paryani_HD_MSN_Caudate, ]
phase_counts_Paryani_HD_MSN_Caudate <- rbind(phase_counts_Paryani_HD_MSN_Caudate, phaseC_genes_notExpParyani_HD_MSN_Caudate)

phaseC_genes_Paryani_CTRL_MSN_Caudate <- intersect(colnames(phase_test), rownames(Paryani_CTRL_MSN_Caudate))
phaseC_genes_notExpParyani_CTRL_MSN_Caudate_names <- setdiff(colnames(phase_test), rownames(Paryani_CTRL_MSN_Caudate))
phaseC_genes_notExpParyani_CTRL_MSN_Caudate <- matrix(data = 0, nrow = length(phaseC_genes_notExpParyani_CTRL_MSN_Caudate_names), ncol = dim(Paryani_CTRL_MSN_Caudate@assays$SCT$data)[2])
rownames(phaseC_genes_notExpParyani_CTRL_MSN_Caudate) <- phaseC_genes_notExpParyani_CTRL_MSN_Caudate_names
colnames(phaseC_genes_notExpParyani_CTRL_MSN_Caudate) <- colnames(Paryani_CTRL_MSN_Caudate@assays$SCT$data)
phase_counts_Paryani_CTRL_MSN_Caudate <- Paryani_CTRL_MSN_Caudate@assays$SCT$data[phaseC_genes_Paryani_CTRL_MSN_Caudate, ]
phase_counts_Paryani_CTRL_MSN_Caudate <- rbind(phase_counts_Paryani_CTRL_MSN_Caudate, phaseC_genes_notExpParyani_CTRL_MSN_Caudate)
```

In [ ]:
```
phase_counts_Paryani_HD_MSN_Caudate <- phase_counts_Paryani_HD_MSN_Caudate[colnames(phase_test), ]
phase_test_Paryani_HD_MSN_Caudate <- data.frame(t(phase_counts_Paryani_HD_MSN_Caudate))
#head(phase_test_Paryani_HD_MSN_Caudate)

phase_counts_Paryani_CTRL_MSN_Caudate <- phase_counts_Paryani_CTRL_MSN_Caudate[colnames(phase_test), ]
phase_test_Paryani_CTRL_MSN_Caudate <- data.frame(t(phase_counts_Paryani_CTRL_MSN_Caudate))
#head(phase_test_Paryani_CTRL_MSN_Caudate)
```

In [ ]:
```
#predict phase
class_thr <- 0.5
predicted_phase_test_Paryani_HD_MSN_Caudate_wprob <- predict(model_phase, n
ewx = as.matrix(phase_test_Paryani_HD_MSN_Caudate), s = lambda_val, type =
"response") #glmnet
predicted_phase_test_Paryani_HD_MSN_Caudate <- rep("A-B", dim(predicted_pha
se_test_Paryani_HD_MSN_Caudate_wprob)[1])
predicted_phase_test_Paryani_HD_MSN_Caudate[which(predicted_phase_test_Pary
ani_HD_MSN_Caudate_wprob > class_thr)] <- "C-D-E" #glmnet
Paryani_HD_MSN_Caudate[["PREDICTED_PHASE"]] <- NA
Paryani_HD_MSN_Caudate@meta.data[rownames(predicted_phase_test_Paryani_HD_M
SN_Caudate_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Paryani_HD_MS
N_Caudate
Paryani_HD_MSN_Caudate[["PROB_PHASE_CDE"]] <- NA
Paryani_HD_MSN_Caudate@meta.data[rownames(predicted_phase_test_Paryani_HD_M
SN_Caudate_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Paryani_HD_MSN
_Caudate_wprob

predicted_phase_test_Paryani_CTRL_MSN_Caudate_wprob <- predict(model_phase,
newx = as.matrix(phase_test_Paryani_CTRL_MSN_Caudate), s = lambda_val, type
= "response") #glmnet
predicted_phase_test_Paryani_CTRL_MSN_Caudate <- rep("A-B", dim(predicted_p
hase_test_Paryani_CTRL_MSN_Caudate_wprob)[1])
predicted_phase_test_Paryani_CTRL_MSN_Caudate[which(predicted_phase_test_Pa
ryani_CTRL_MSN_Caudate_wprob > class_thr)] <- "C-D-E" #glmnet
Paryani_CTRL_MSN_Caudate[["PREDICTED_PHASE"]] <- NA
Paryani_CTRL_MSN_Caudate@meta.data[rownames(predicted_phase_test_Paryani_CT
RL_MSN_Caudate_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Paryani_C
TRL_MSN_Caudate
Paryani_CTRL_MSN_Caudate[["PROB_PHASE_CDE"]] <- NA
Paryani_CTRL_MSN_Caudate@meta.data[rownames(predicted_phase_test_Paryani_CT
RL_MSN_Caudate_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Paryani_CT
RL_MSN_Caudate_wprob
```

In [ ]:
```
table(predicted_phase_test_Paryani_HD_MSN_Caudate)
summary(predicted_phase_test_Paryani_HD_MSN_Caudate_wprob)
colnames(Paryani_HD_MSN_Caudate@meta.data)

table(predicted_phase_test_Paryani_CTRL_MSN_Caudate)
summary(predicted_phase_test_Paryani_CTRL_MSN_Caudate_wprob)
colnames(Paryani_CTRL_MSN_Caudate@meta.data)
```

In [ ]:
```r
#do plots
DimPlot(Paryani_HD_MSN_Caudate, group.by = "PREDICTED_PHASE", reduction =
"harmony", pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Caudate_PREDPHASE.pdf", width = 8, height = 8)
FeaturePlot(Paryani_HD_MSN_Caudate, features = "PROB_PHASE_CDE", cols = c
("blue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Paryani_HD_MSN_Caudate_PROBPREDPHASE.pdf", width = 8, height =
8)

DimPlot(Paryani_CTRL_MSN_Caudate, group.by = "PREDICTED_PHASE", reduction =
"harmony", pt.size = 1)
ggsave("UMAP_Paryani_CTRL_MSN_Caudate_PREDPHASE.pdf", width = 8, height =
8)
FeaturePlot(Paryani_CTRL_MSN_Caudate, features = "PROB_PHASE_CDE", cols = c
("blue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Paryani_CTRL_MSN_Caudate_PROBPREDPHASE.pdf", width = 8, height
= 8)
DimPlot(Paryani_CTRL_MSN_Caudate, group.by = "Donor", reduction = "harmon
y", pt.size = 1)
ggsave("UMAP_Paryani_CTRL_MSN_Caudate_DONOR.pdf", width = 8, height = 8)
```

In [ ]:
```r
#plot the fraction of SPN in C-D-E phase vs the number of CAG in the germli
ne
fract_CDE_tmp_Caudate <- lapply(split(Paryani_HD_MSN_Caudate@meta.data, Par
yani_HD_MSN_Caudate@meta.data$CAG), function(x) {
    num_SPN <- dim(x)[1]
    num_CDE <- length(which(x$PREDICTED_PHASE == "C-D-E"))
    fraction_CDE <- num_CDE/num_SPN
    num_CAG <- gsub(x = x$CAG[1], pattern = "-.*", replacement = "")
    #cat(sprintf("Fraction MSNs in C-D-E phase: %.2f\n", fraction_CDE*100))
    return(rbind(fraction_CDE, num_SPN, num_CAG))
})
fract_CDE_Caudate <- t(do.call(cbind, fract_CDE_tmp_Caudate))
fract_CDE_Caudate <- data.frame(num_CAG_germ = as.numeric(fract_CDE_Caudate
[, 3]), fract_CDE = as.numeric(fract_CDE_Caudate[, 1]), num_SPN = as.numeri
c(fract_CDE_Caudate[, 2]))
#cor_numCAGGerm_fractCDE_Caudate <- cor(x = fract_CDE_Caudate$num_CAG_germ,
y = fract_CDE_Caudate$fract_CDE, method = "pearson", use="complete.obs")

cor_numCAGGerm_fractCDE_Caudate <- weighted.cor(x = fract_CDE_Caudate$num_C
AG_germ, y = fract_CDE_Caudate$fract_CDE, weights = fract_CDE_Caudate$num_S
PN, method = "pearson", na.rm = TRUE)

print(cor_numCAGGerm_fractCDE_Caudate)
#plot(x = fract_CDE_Caudate$num_CAG_germ, y = fract_CDE_Caudate$fract_CDE)

ggplot(fract_CDE_Caudate, aes(x=num_CAG_germ, y=100*fract_CDE)) +
  geom_point(col = "blue", aes(size = num_SPN))+
  geom_smooth(method=lm, se = TRUE, mapping = aes(weight = num_SPN)) +
  theme_classic() +
  xlab("Num. CAG germline") +
  ylab("Fraction cells in C-D-E phase (%)")
  #+ ggtitle(sprintf("Sp. corr = %.2f", cor_numCAGGerm_fractCDE_Caudate))
  #+ xlim(c(36, 70)) + ylim(c(0, 1))
ggsave("Paryani_NumCAGGermline_FractCDE_Caudate.pdf", width = 8, height =
8)
```

In [ ]:
```r
#create metadata for plotting
Paryani_Caudate_Grade_CDE_tmp <- lapply(split(Paryani_HD_MSN_Caudate@meta.data, Paryani_HD_MSN_Caudate@meta.data$Donor), function(x) {
    GRADE <- factor(unique(x$Grade), levels = c("HD1", "HD2", "HD3", "HD4", "HDJ"))
    SN <- unique(x$Donor)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    tmp <- lapply(split(x, x$sub_type_4), function(y) {
        NUM_MSN_CT <- length(y$PREDICTED_PHASE)
        NUM_MSN_CT_SQ <- NUM_MSN_CT**2
        CELL_TYPE <- y$sub_type_4
        if (length(unique(y$PREDICTED_PHASE)) > 1) {
          val <- 100*table(y$PREDICTED_PHASE)[2]/(table(y$PREDICTED_PHASE)[1] + table(y$PREDICTED_PHASE)[2])
        } else {
          val <- 0
        }
        names(val) <- "Fract_CDE";
        Fract_CDE <- rep(val, NUM_MSN_CT_SQ);
    return(data.frame(NUM_MSN_CT_SQ, CELL_TYPE, Fract_CDE))})
    df <- do.call(rbind, tmp)
    #print(df)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    #Fract_CDE <- 100*length(which(x$PREDICTED_PHASE == "C-D-E"))/length(x$PREDICTED_PHASE)
    #CELL_TYPE <- x$sub_type_4
    #return(df)
    #return(data.frame(SAMPLE = rep(SN, df$NUM_MSN_CT_SQ), GRADE = rep(GRADE, df$NUM_MSN_CT_SQ), CELL_TYPE = df$CELL_TYPE, Fract_CDE = df$Fract_CDE, NUM_MSN_SQ=df$NUM_MSN_CT_SQ))
    return(data.frame(SAMPLE = SN, GRADE = GRADE, CELL_TYPE = df$CELL_TYPE, Fract_CDE = df$Fract_CDE, NUM_MSN_SQ=df$NUM_MSN_CT_SQ))
})
Paryani_Caudate_Grade_CDE <- do.call(rbind, Paryani_Caudate_Grade_CDE_tmp)
Paryani_Caudate_Grade_CDE <- Paryani_Caudate_Grade_CDE[order(Paryani_Caudate_Grade_CDE$GRADE), ]
Paryani_Caudate_Grade_CDE$SAMPLE <- factor(Paryani_Caudate_Grade_CDE$SAMPLE, levels = unique(Paryani_Caudate_Grade_CDE$SAMPLE))
head(Paryani_Caudate_Grade_CDE)
```

In [ ]:
```r
uni_Paryani_Caudate_Grade_CDE <- unique(Paryani_Caudate_Grade_CDE)
uni_Paryani_Caudate_Grade_CDE$NUM_SPN <- sqrt(uni_Paryani_Caudate_Grade_CDE$NUM_MSN_SQ)
#uni_Paryani_Caudate_Grade_CDE
```

In [ ]:

```
#add metadata
Fract_SPN_Caudate <- c(num_cells_Paryani["T-4285", "num_dSPN1_caudate"]/num
_cells_Paryani["T-4285", "num_cells_caudate"],
                       num_cells_Paryani["T-4285", "num_dSPN2_caudate"]/num
_cells_Paryani["T-4285", "num_cells_caudate"],
                       num_cells_Paryani["T-4285", "num_iSPN1_caudate"]/num
_cells_Paryani["T-4285", "num_cells_caudate"],
                       num_cells_Paryani["T-4285", "num_iSPN2_caudate"]/num
_cells_Paryani["T-4285", "num_cells_caudate"],

                       num_cells_Paryani["T-5266", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5266", "num_cells_caudate"],
                       #num_cells_Paryani["T-5266", "num_dSPN2_caudate"]/nu
m_cells_Paryani["T-5266", "num_cells_caudate"],
                       num_cells_Paryani["T-5266", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5266", "num_cells_caudate"],
                       num_cells_Paryani["T-5266", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5266", "num_cells_caudate"],

                       num_cells_Paryani["T-5798", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5798", "num_cells_caudate"],
                       num_cells_Paryani["T-5798", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5798", "num_cells_caudate"],
                       num_cells_Paryani["T-5798", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5798", "num_cells_caudate"],
                       num_cells_Paryani["T-5798", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5798", "num_cells_caudate"],

                       num_cells_Paryani["T-5263", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5263", "num_cells_caudate"],
                       num_cells_Paryani["T-5263", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5263", "num_cells_caudate"],
                       num_cells_Paryani["T-5263", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5263", "num_cells_caudate"],
                       num_cells_Paryani["T-5263", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5263", "num_cells_caudate"],

                       num_cells_Paryani["T-5286", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5286", "num_cells_caudate"],
                       num_cells_Paryani["T-5286", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5286", "num_cells_caudate"],
                       num_cells_Paryani["T-5286", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5286", "num_cells_caudate"],
                       num_cells_Paryani["T-5286", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5286", "num_cells_caudate"],

                       num_cells_Paryani["T-5693", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5693", "num_cells_caudate"],
                       num_cells_Paryani["T-5693", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5693", "num_cells_caudate"],
                       num_cells_Paryani["T-5693", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5693", "num_cells_caudate"],
                       num_cells_Paryani["T-5693", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5693", "num_cells_caudate"],

                       num_cells_Paryani["T-4161", "num_dSPN1_caudate"]/num
_cells_Paryani["T-4161", "num_cells_caudate"],
                       num_cells_Paryani["T-4161", "num_dSPN2_caudate"]/num
_cells_Paryani["T-4161", "num_cells_caudate"],
                       num_cells_Paryani["T-4161", "num_iSPN1_caudate"]/num
```

```
_cells_Paryani["T-4161", "num_cells_caudate"],
                        num_cells_Paryani["T-4161", "num_iSPN2_caudate"]/num
_cells_Paryani["T-4161", "num_cells_caudate"],

                        num_cells_Paryani["T-5493", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5493", "num_cells_caudate"],
                        num_cells_Paryani["T-5493", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5493", "num_cells_caudate"],
                        num_cells_Paryani["T-5493", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5493", "num_cells_caudate"],
                        num_cells_Paryani["T-5493", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5493", "num_cells_caudate"],

                        num_cells_Paryani["T-5575", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5575", "num_cells_caudate"],
                        num_cells_Paryani["T-5575", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5575", "num_cells_caudate"],
                        num_cells_Paryani["T-5575", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5575", "num_cells_caudate"],
                        num_cells_Paryani["T-5575", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5575", "num_cells_caudate"],

                        num_cells_Paryani["T-4273", "num_dSPN1_caudate"]/num
_cells_Paryani["T-4273", "num_cells_caudate"],
                        #num_cells_Paryani["T-4273", "num_dSPN2_caudate"]/nu
m_cells_Paryani["T-4273", "num_cells_caudate"],
                        num_cells_Paryani["T-4273", "num_iSPN1_caudate"]/num
_cells_Paryani["T-4273", "num_cells_caudate"],
                        num_cells_Paryani["T-4273", "num_iSPN2_caudate"]/num
_cells_Paryani["T-4273", "num_cells_caudate"],

                        num_cells_Paryani["T-4638", "num_dSPN1_caudate"]/num
_cells_Paryani["T-4638", "num_cells_caudate"],
                        #num_cells_Paryani["T-4638", "num_dSPN2_caudate"]/nu
m_cells_Paryani["T-4638", "num_cells_caudate"],
                        num_cells_Paryani["T-4638", "num_iSPN1_caudate"]/num
_cells_Paryani["T-4638", "num_cells_caudate"],
                        num_cells_Paryani["T-4638", "num_iSPN2_caudate"]/num
_cells_Paryani["T-4638", "num_cells_caudate"],

                        num_cells_Paryani["T-5534", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5534", "num_cells_caudate"],
                        num_cells_Paryani["T-5534", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5534", "num_cells_caudate"],
                        num_cells_Paryani["T-5534", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5534", "num_cells_caudate"],
                        num_cells_Paryani["T-5534", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5534", "num_cells_caudate"],

                        num_cells_Paryani["T-5714", "num_dSPN1_caudate"]/num
_cells_Paryani["T-5714", "num_cells_caudate"],
                        num_cells_Paryani["T-5714", "num_dSPN2_caudate"]/num
_cells_Paryani["T-5714", "num_cells_caudate"],
                        num_cells_Paryani["T-5714", "num_iSPN1_caudate"]/num
_cells_Paryani["T-5714", "num_cells_caudate"],
                        num_cells_Paryani["T-5714", "num_iSPN2_caudate"]/num
_cells_Paryani["T-5714", "num_cells_caudate"])
#Fract_SPN_Caudate
uni_Paryani_Caudate_Grade_CDE$Fract_SPN <- Fract_SPN_Caudate
uni_Paryani_Caudate_Grade_CDE
```

In [ ]:
```
# ggplot(Paryani_Caudate_Grade_CDE, aes(x = SAMPLE, y = Fract_CDE, color = CELL_TYPE)) +
# geom_boxplot(varwidth = TRUE, lwd = 5, position = position_dodge2(preserve = "single")) +
# scale_color_manual(values = c("#F8766D", "#7CAE00", "#00BFC4", "#C77CFF")) +
# theme_classic() +
# xlab("SAMPLE") +
# ylim(c(0, 100)) +
# ylab("Fraction cells in C-D-E phase (%)")
# ggsave("Paryani_Caudate_CELLTYPE_vs_fraction_SPN_CDE.pdf", width = 8, height = 8)

#plot fraction of SPNs in C-D-E phase by HD grade
ggplot(uni_Paryani_Caudate_Grade_CDE, aes(x = GRADE, y = Fract_CDE, color = CELL_TYPE)) +
geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(width = 0.1, height = 0)) +
scale_color_manual(values =c("#F8766D", "#7CAE00", "#00BFC4", "#C77CFF")) +
theme_classic() +
xlab("HD GRADE") +
#ylim(c(0, 100)) +
ylab("Fraction cells in C-D-E phase (%)")
ggsave("Paryani_Caudate_GRADE_vs_fraction_SPN_CDE.pdf", width = 6, height = 6)
```

In [ ]:
```
# ggplot(uni_Paryani_Caudate_Grade_CDE[which(uni_Paryani_Caudate_Grade_CDE$NUM_SPN > 30), ], aes(x = GRADE, y = Fract_CDE, color = CELL_TYPE)) +
# geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(width = 0.1, height = 0)) +
# scale_color_manual(values =c("#F8766D", "#7CAE00", "#00BFC4", "#C77CFF")) +
# theme_classic() +
# xlab("HD GRADE") +
# #ylim(c(0, 100)) +
# ylab("Fraction cells in C-D-E phase (%)")
# ggsave("Paryani_Caudate_GRADE_vs_fraction_SPN_CDE_gt30.pdf", width = 6, height = 6)
```

Paryani_Accumbens_only

In [ ]:
```
#Run SCTransform for HD donors
Paryani_HD_MSN_Accumbens <- SCTransform(Paryani_HD_MSN_Accumbens, vars.to.r
egress = c("nCount_RNA", "nFeature_RNA"))
#Run PCA and UMAP for HD donors
Paryani_HD_MSN_Accumbens <- RunPCA(Paryani_HD_MSN_Accumbens, features = Var
iableFeatures(object = Paryani_HD_MSN_Accumbens), npcs = 50, reduction.name
= "pca")
ElbowPlot(Paryani_HD_MSN, reduction = "pca", ndims = 50)
num_dims <- 40
Paryani_HD_MSN_Accumbens <- RunUMAP(Paryani_HD_MSN_Accumbens, dims = 1:num_
dims, reduction = "pca", reduction.name = "umap", reduction.key = "umap")
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "Batch", reduction = "umap", p
t.size = 1)
Paryani_HD_MSN_Accumbens <- RunHarmony(object = Paryani_HD_MSN_Accumbens, r
eduction.use = "pca", group.by.vars = "Batch",
                              reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Paryani_HD_MSN_Accumbens <- RunUMAP(Paryani_HD_MSN_Accumbens, dims = 1:num_
dims, reduction = "harmonyPca", reduction.name = "harmony", reduction.key =
"harmony")
```

In [ ]:
```
#Run SCTransform for CTRL donors
Paryani_CTRL_MSN_Accumbens <- SCTransform(Paryani_CTRL_MSN_Accumbens, vars.
to.regress = c("nCount_RNA", "nFeature_RNA"))
#Run PCA and UMAP for CTRL donors
Paryani_CTRL_MSN_Accumbens <- RunPCA(Paryani_CTRL_MSN_Accumbens, features =
VariableFeatures(object = Paryani_CTRL_MSN_Accumbens), npcs = 50, reductio
n.name = "pca")
ElbowPlot(Paryani_CTRL_MSN, reduction = "pca", ndims = 50)
num_dims <- 40
Paryani_CTRL_MSN_Accumbens <- RunUMAP(Paryani_CTRL_MSN_Accumbens, dims = 1:
num_dims, reduction = "pca", reduction.name = "umap", reduction.key = "uma
p")
DimPlot(Paryani_CTRL_MSN_Accumbens, group.by = "Batch", reduction = "umap",
pt.size = 3)
Paryani_CTRL_MSN_Accumbens <- RunHarmony(object = Paryani_CTRL_MSN_Accumben
s, reduction.use = "pca", group.by.vars = "Batch",
                              reduction.save = "harmonyPca", assay = "SCT", ve
rbose = FALSE, normalization.method = "SCT")
Paryani_CTRL_MSN_Accumbens <- RunUMAP(Paryani_CTRL_MSN_Accumbens, dims = 1:
num_dims, reduction = "harmonyPca", reduction.name = "harmony", reduction.k
ey = "harmony")
```

In [ ]:

```
#do plots
FeaturePlot(Paryani_HD_MSN_Accumbens, reduction = "harmony", features = "nC
ount_RNA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_NCOUNTS.pdf", width = 8, height = 8)
FeaturePlot(Paryani_HD_MSN_Accumbens, reduction = "harmony", features = "nF
eature_RNA", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_NFEATURES.pdf", width = 8, height =
8)
Paryani_HD_MSN_Accumbens@meta.data$GERM_EXPCAGLENGTH <- as.numeric(gsub(x =
Paryani_HD_MSN_Accumbens@meta.data$CAG, pattern = "-.*", replacement = ""))
FeaturePlot(Paryani_HD_MSN_Accumbens, reduction = "harmony", features = "GE
RM_EXPCAGLENGTH", cols = c("blue", "red"), pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_GERM_EXPCAGLENGTH.pdf", width = 8, he
ight = 8)
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "sub_type_4", reduction = "har
mony", pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_CELLTYPE.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "CAG", reduction = "harmony",
pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_CAG.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "Batch", reduction = "harmon
y", pt.size = 1)
ggsave("UMAP_HD_Paryani_MSN_Accumbens_BATCH.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "Region", reduction = "harmon
y", pt.size = 1)
ggsave("UMAP_Paryani_MSN_Accumbens_REGION.pdf", width = 8, height = 8)
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "Donor", reduction = "harmon
y", pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_DONOR.pdf", width = 8, height = 8)
Paryani_HD_MSN_Accumbens@meta.data$Grade <- factor(Paryani_HD_MSN_Accumbens
@meta.data$Grade, levels = c("HDJ", "HD4", "HD3", "HD2", "HD1", "Control"))
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "Grade", reduction = "harmon
y", pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_GRADE.pdf", width = 8, height = 8)
```

In [ ]:
```
#extract phase C genes in the model that are also expressed in the dataset
phaseC_genes_Paryani_HD_MSN_Accumbens <- intersect(colnames(phase_test), ro
wnames(Paryani_HD_MSN_Accumbens))
phaseC_genes_notExpParyani_HD_MSN_Accumbens_names <- setdiff(colnames(phase
_test), rownames(Paryani_HD_MSN_Accumbens))
phaseC_genes_notExpParyani_HD_MSN_Accumbens <- matrix(data = 0, nrow = leng
th(phaseC_genes_notExpParyani_HD_MSN_Accumbens_names), ncol = dim(Paryani_H
D_MSN_Accumbens@assays$SCT$data)[2])
rownames(phaseC_genes_notExpParyani_HD_MSN_Accumbens) <- phaseC_genes_notEx
pParyani_HD_MSN_Accumbens_names
colnames(phaseC_genes_notExpParyani_HD_MSN_Accumbens) <- colnames(Paryani_H
D_MSN_Accumbens@assays$SCT$data)
phase_counts_Paryani_HD_MSN_Accumbens <- Paryani_HD_MSN_Accumbens@assays$SC
T$data[phaseC_genes_Paryani_HD_MSN_Accumbens, ]
phase_counts_Paryani_HD_MSN_Accumbens <- rbind(phase_counts_Paryani_HD_MSN_
Accumbens, phaseC_genes_notExpParyani_HD_MSN_Accumbens)

phaseC_genes_Paryani_CTRL_MSN_Accumbens <- intersect(colnames(phase_test),
rownames(Paryani_CTRL_MSN_Accumbens))
phaseC_genes_notExpParyani_CTRL_MSN_Accumbens_names <- setdiff(colnames(pha
se_test), rownames(Paryani_CTRL_MSN_Accumbens))
phaseC_genes_notExpParyani_CTRL_MSN_Accumbens <- matrix(data = 0, nrow = le
ngth(phaseC_genes_notExpParyani_CTRL_MSN_Accumbens_names), ncol = dim(Parya
ni_CTRL_MSN_Accumbens@assays$SCT$data)[2])
rownames(phaseC_genes_notExpParyani_CTRL_MSN_Accumbens) <- phaseC_genes_not
ExpParyani_CTRL_MSN_Accumbens_names
colnames(phaseC_genes_notExpParyani_CTRL_MSN_Accumbens) <- colnames(Paryani
_CTRL_MSN_Accumbens@assays$SCT$data)
phase_counts_Paryani_CTRL_MSN_Accumbens <- Paryani_CTRL_MSN_Accumbens@assay
s$SCT$data[phaseC_genes_Paryani_CTRL_MSN_Accumbens, ]
phase_counts_Paryani_CTRL_MSN_Accumbens <- rbind(phase_counts_Paryani_CTRL_
MSN_Accumbens, phaseC_genes_notExpParyani_CTRL_MSN_Accumbens)
```

In [ ]:
```
phase_counts_Paryani_HD_MSN_Accumbens <- phase_counts_Paryani_HD_MSN_Accumb
ens[colnames(phase_test), ]
phase_test_Paryani_HD_MSN_Accumbens <- data.frame(t(phase_counts_Paryani_HD
_MSN_Accumbens))
#head(phase_test_Paryani_HD_MSN_Accumbens)

phase_counts_Paryani_CTRL_MSN_Accumbens <- phase_counts_Paryani_CTRL_MSN_Ac
cumbens[colnames(phase_test), ]
phase_test_Paryani_CTRL_MSN_Accumbens <- data.frame(t(phase_counts_Paryani_
CTRL_MSN_Accumbens))
#head(phase_test_Paryani_CTRL_MSN_Accumbens)
```

In [ ]:
```
#predict phase
class_thr <- 0.5
predicted_phase_test_Paryani_HD_MSN_Accumbens_wprob <- predict(model_phase, newx = as.matrix(phase_test_Paryani_HD_MSN_Accumbens), s = lambda_val, type = "response") #glmnet
predicted_phase_test_Paryani_HD_MSN_Accumbens <- rep("A-B", dim(predicted_phase_test_Paryani_HD_MSN_Accumbens_wprob)[1])
predicted_phase_test_Paryani_HD_MSN_Accumbens[which(predicted_phase_test_Paryani_HD_MSN_Accumbens_wprob > class_thr)] <- "C-D-E" #glmnet
Paryani_HD_MSN_Accumbens[["PREDICTED_PHASE"]] <- NA
Paryani_HD_MSN_Accumbens@meta.data[rownames(predicted_phase_test_Paryani_HD_MSN_Accumbens_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Paryani_HD_MSN_Accumbens
Paryani_HD_MSN_Accumbens[["PROB_PHASE_CDE"]] <- NA
Paryani_HD_MSN_Accumbens@meta.data[rownames(predicted_phase_test_Paryani_HD_MSN_Accumbens_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Paryani_HD_MSN_Accumbens_wprob

predicted_phase_test_Paryani_CTRL_MSN_Accumbens_wprob <- predict(model_phase, newx = as.matrix(phase_test_Paryani_CTRL_MSN_Accumbens), s = lambda_val, type = "response") #glmnet
predicted_phase_test_Paryani_CTRL_MSN_Accumbens <- rep("A-B", dim(predicted_phase_test_Paryani_CTRL_MSN_Accumbens_wprob)[1])
predicted_phase_test_Paryani_CTRL_MSN_Accumbens[which(predicted_phase_test_Paryani_CTRL_MSN_Accumbens_wprob > class_thr)] <- "C-D-E" #glmnet
Paryani_CTRL_MSN_Accumbens[["PREDICTED_PHASE"]] <- NA
Paryani_CTRL_MSN_Accumbens@meta.data[rownames(predicted_phase_test_Paryani_CTRL_MSN_Accumbens_wprob), "PREDICTED_PHASE"] <- predicted_phase_test_Paryani_CTRL_MSN_Accumbens
Paryani_CTRL_MSN_Accumbens[["PROB_PHASE_CDE"]] <- NA
Paryani_CTRL_MSN_Accumbens@meta.data[rownames(predicted_phase_test_Paryani_CTRL_MSN_Accumbens_wprob), "PROB_PHASE_CDE"] <- predicted_phase_test_Paryani_CTRL_MSN_Accumbens_wprob
```

In [ ]:
```
table(predicted_phase_test_Paryani_HD_MSN_Accumbens)
summary(predicted_phase_test_Paryani_HD_MSN_Accumbens_wprob)

table(predicted_phase_test_Paryani_CTRL_MSN_Accumbens)
summary(predicted_phase_test_Paryani_CTRL_MSN_Accumbens_wprob)
```

In [ ]:
```
#do plots
DimPlot(Paryani_HD_MSN_Accumbens, group.by = "PREDICTED_PHASE", reduction =
"harmony", pt.size = 1)
ggsave("UMAP_Paryani_HD_MSN_Accumbens_PREDPHASE.pdf", width = 8, height =
8)
FeaturePlot(Paryani_HD_MSN_Accumbens, features = "PROB_PHASE_CDE", cols = c
("blue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Paryani_HD_MSN_Accumbens_PROBPREDPHASE.pdf", width = 8, height
= 8)

DimPlot(Paryani_CTRL_MSN_Accumbens, group.by = "PREDICTED_PHASE", reduction
= "harmony", pt.size = 1)
ggsave("UMAP_Paryani_CTRL_MSN_Accumbens_PREDPHASE.pdf", width = 8, height =
8)
FeaturePlot(Paryani_CTRL_MSN_Accumbens, features = "PROB_PHASE_CDE", cols =
c("blue", "red"), pt.size = 1, reduction = "harmony")
ggsave("UMAP_Paryani_CTRL_MSN_Accumbens_PROBPREDPHASE.pdf", width = 8, heig
ht = 8)
DimPlot(Paryani_CTRL_MSN_Accumbens, group.by = "Donor", reduction = "harmon
y", pt.size = 1)
ggsave("UMAP_Paryani_CTRL_MSN_Accumbens_DONOR.pdf", width = 8, height = 8)
```

In [ ]:
```
#plot the fraction of SPN in C-D-E phase vs the number of CAG in the germli
ne
fract_CDE_tmp_Accumbens <- lapply(split(Paryani_HD_MSN_Accumbens@meta.data,
Paryani_HD_MSN_Accumbens@meta.data$CAG), function(x) {
    num_SPN <- dim(x)[1]
    num_CDE <- length(which(x$PREDICTED_PHASE == "C-D-E"))
    fraction_CDE <- num_CDE/num_SPN
    num_CAG <- gsub(x = x$CAG[1], pattern = "-.*", replacement = "")
    #cat(sprintf("Fraction MSNs in C-D-E phase: %.2f\n", fraction_CDE*100))
    return(rbind(fraction_CDE, num_SPN, num_CAG))
})
fract_CDE_Accumbens <- t(do.call(cbind, fract_CDE_tmp_Accumbens))
fract_CDE_Accumbens <- data.frame(num_CAG_germ = as.numeric(fract_CDE_Accum
bens[, 3]), fract_CDE = as.numeric(fract_CDE_Accumbens[, 1]), num_SPN = as.
numeric(fract_CDE_Accumbens[, 2]))
#cor_numCAGGerm_fractCDE_Accumbens <- cor(x = fract_CDE_Accumbens$num_CAG_g
erm, y = fract_CDE_Accumbens$fract_CDE, method = "pearson", use="complete.o
bs")

cor_numCAGGerm_fractCDE_Accumbens <- weighted.cor(x = fract_CDE_Accumbens$n
um_CAG_germ, y = fract_CDE_Accumbens$fract_CDE, weights = fract_CDE_Accumbe
ns$num_SPN, method = "pearson", na.rm = TRUE)
print(cor_numCAGGerm_fractCDE_Accumbens)

#plot(x = fract_CDE_Accumbens$num_CAG_germ, y = fract_CDE_Accumbens$fract_C
DE)

ggplot(fract_CDE_Accumbens, aes(x=num_CAG_germ, y=100*fract_CDE)) +
  geom_point(col = "blue", aes(size = num_SPN))+
  geom_smooth(method=lm, se = TRUE, mapping = aes(weight = num_SPN)) +
  theme_classic() +
  xlab("Num. CAG germline") +
  ylab("Fraction cells in C-D-E phase (%)")
  #+ ggtitle(sprintf("Sp. corr = %.2f", cor_numCAGGerm_fractCDE_Accumbens))
  #+ xlim(c(36, 70)) + ylim(c(0, 1))
ggsave("Paryani_NumCAGGermline_FractCDE_Accumbens.pdf", width = 8, height =
8)
```

In [ ]:
```
#create metadata for plotting
Paryani_Accumbens_Grade_CDE_tmp <- lapply(split(Paryani_HD_MSN_Accumbens@meta.data, Paryani_HD_MSN_Accumbens@meta.data$Donor), function(x) {
    GRADE <- factor(unique(x$Grade), levels = c("HD1", "HD2", "HD3", "HD4", "HDJ"))
    SN <- unique(x$Donor)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    tmp <- lapply(split(x, x$sub_type_4), function(y) {
        NUM_MSN_CT <- length(y$PREDICTED_PHASE)
        NUM_MSN_CT_SQ <- NUM_MSN_CT**2
        CELL_TYPE <- y$sub_type_4
        if (length(unique(y$PREDICTED_PHASE)) > 1) {
          val <- 100*table(y$PREDICTED_PHASE)[2]/(table(y$PREDICTED_PHASE)[1] + table(y$PREDICTED_PHASE)[2])
        } else {
          val <- 0
        }
        names(val) <- "Fract_CDE";
        Fract_CDE <- rep(val, NUM_MSN_CT_SQ);
    return(data.frame(NUM_MSN_CT_SQ, CELL_TYPE, Fract_CDE))})
    df <- do.call(rbind, tmp)
    #print(df)
    #NUM_MSN <- length(x$PREDICTED_PHASE)
    #Fract_CDE <- 100*length(which(x$PREDICTED_PHASE == "C-D-E"))/length(x$PREDICTED_PHASE)
    #CELL_TYPE <- x$sub_type_4
    #return(df)
    #return(data.frame(SAMPLE = rep(SN, df$NUM_MSN_CT_SQ), GRADE = rep(GRADE, df$NUM_MSN_CT_SQ), CELL_TYPE = df$CELL_TYPE, Fract_CDE = df$Fract_CDE, NUM_MSN_SQ=df$NUM_MSN_CT_SQ))
    return(data.frame(SAMPLE = SN, GRADE = GRADE, CELL_TYPE = df$CELL_TYPE, Fract_CDE = df$Fract_CDE, NUM_MSN_SQ=df$NUM_MSN_CT_SQ))
})
Paryani_Accumbens_Grade_CDE <- do.call(rbind, Paryani_Accumbens_Grade_CDE_tmp)
Paryani_Accumbens_Grade_CDE <- Paryani_Accumbens_Grade_CDE[order(Paryani_Accumbens_Grade_CDE$GRADE), ]
Paryani_Accumbens_Grade_CDE$SAMPLE <- factor(Paryani_Accumbens_Grade_CDE$SAMPLE, levels = unique(Paryani_Accumbens_Grade_CDE$SAMPLE))
#head(Paryani_Accumbens_Grade_CDE)
```

In [ ]:
```
uni_Paryani_Accumbens_Grade_CDE <- unique(Paryani_Accumbens_Grade_CDE)
uni_Paryani_Accumbens_Grade_CDE$NUM_SPN <- sqrt(uni_Paryani_Accumbens_Grade_CDE$NUM_MSN_SQ)
#uni_Paryani_Accumbens_Grade_CDE
#uni_Paryani_Caudate_Grade_CDE[which(uni_Paryani_Accumbens_Grade_CDE$CELL_TYPE == "iSPN_2"), ]
```

In [ ]:

```
#add metadata
Fract_SPN_Accumbens <- c(num_cells_Paryani["T-4285", "num_dSPN1_accumben
s"]/num_cells_Paryani["T-4285", "num_cells_accumbens"],
                         num_cells_Paryani["T-4285", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-4285", "num_cells_accumbens"],
                         num_cells_Paryani["T-4285", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-4285", "num_cells_accumbens"],
                         num_cells_Paryani["T-4285", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-4285", "num_cells_accumbens"],

                         num_cells_Paryani["T-5266", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5266", "num_cells_accumbens"],
                         num_cells_Paryani["T-5266", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5266", "num_cells_accumbens"],
                         num_cells_Paryani["T-5266", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5266", "num_cells_accumbens"],
                         num_cells_Paryani["T-5266", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5266", "num_cells_accumbens"],

                         num_cells_Paryani["T-5263", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5263", "num_cells_accumbens"],
                         num_cells_Paryani["T-5263", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5263", "num_cells_accumbens"],
                         num_cells_Paryani["T-5263", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5263", "num_cells_accumbens"],
                         num_cells_Paryani["T-5263", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5263", "num_cells_accumbens"],

                         num_cells_Paryani["T-5608", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5608", "num_cells_accumbens"],
                         num_cells_Paryani["T-5608", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5608", "num_cells_accumbens"],
                         num_cells_Paryani["T-5608", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5608", "num_cells_accumbens"],
                         num_cells_Paryani["T-5608", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5608", "num_cells_accumbens"],

                         num_cells_Paryani["T-5693", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5693", "num_cells_accumbens"],
                         num_cells_Paryani["T-5693", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5693", "num_cells_accumbens"],
                         num_cells_Paryani["T-5693", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5693", "num_cells_accumbens"],
                         num_cells_Paryani["T-5693", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5693", "num_cells_accumbens"],

                         num_cells_Paryani["T-4161", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-4161", "num_cells_accumbens"],
                         num_cells_Paryani["T-4161", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-4161", "num_cells_accumbens"],
                         num_cells_Paryani["T-4161", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-4161", "num_cells_accumbens"],
                         num_cells_Paryani["T-4161", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-4161", "num_cells_accumbens"],

                         num_cells_Paryani["T-5109", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5109", "num_cells_accumbens"],
                         num_cells_Paryani["T-5109", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5109", "num_cells_accumbens"],
                         num_cells_Paryani["T-5109", "num_iSPN1_accumbens"]/n
```

```
um_cells_Paryani["T-5109", "num_cells_accumbens"],
                        num_cells_Paryani["T-5109", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5109", "num_cells_accumbens"],

                        num_cells_Paryani["T-5493", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5493", "num_cells_accumbens"],
                        num_cells_Paryani["T-5493", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5493", "num_cells_accumbens"],
                        num_cells_Paryani["T-5493", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5493", "num_cells_accumbens"],
                        num_cells_Paryani["T-5493", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5493", "num_cells_accumbens"],

                        num_cells_Paryani["T-5575", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5575", "num_cells_accumbens"],
                        #num_cells_Paryani["T-5575", "num_dSPN2_accumbens"]/
num_cells_Paryani["T-5575", "num_cells_accumbens"],
                        num_cells_Paryani["T-5575", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5575", "num_cells_accumbens"],
                        num_cells_Paryani["T-5575", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5575", "num_cells_accumbens"],

                        num_cells_Paryani["T-4638", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-4638", "num_cells_accumbens"],
                        num_cells_Paryani["T-4638", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-4638", "num_cells_accumbens"],
                        num_cells_Paryani["T-4638", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-4638", "num_cells_accumbens"],
                        num_cells_Paryani["T-4638", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-4638", "num_cells_accumbens"],

                        num_cells_Paryani["T-5534", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5534", "num_cells_accumbens"],
                        num_cells_Paryani["T-5534", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5534", "num_cells_accumbens"],
                        num_cells_Paryani["T-5534", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5534", "num_cells_accumbens"],
                        num_cells_Paryani["T-5534", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5534", "num_cells_accumbens"],

                        num_cells_Paryani["T-5714", "num_dSPN1_accumbens"]/n
um_cells_Paryani["T-5714", "num_cells_accumbens"],
                        num_cells_Paryani["T-5714", "num_dSPN2_accumbens"]/n
um_cells_Paryani["T-5714", "num_cells_accumbens"],
                        num_cells_Paryani["T-5714", "num_iSPN1_accumbens"]/n
um_cells_Paryani["T-5714", "num_cells_accumbens"],
                        num_cells_Paryani["T-5714", "num_iSPN2_accumbens"]/n
um_cells_Paryani["T-5714", "num_cells_accumbens"])
#Fract_SPN_accumbens
uni_Paryani_Accumbens_Grade_CDE$Fract_SPN <- Fract_SPN_Accumbens
uni_Paryani_Accumbens_Grade_CDE
```

In [ ]:
```
# ggplot(Paryani_Accumbens_Grade_CDE, aes(x = SAMPLE, y = Fract_CDE, color
= CELL_TYPE)) +
# geom_boxplot(varwidth = TRUE, lwd = 5, position = position_dodge2(preserv
e = "single")) +
# scale_color_manual(values = c("#F8766D", "#7CAE00", "#00BFC4", "#C77CF
F")) +
# theme_classic() +
# xlab("SAMPLE") +
# ylim(c(0, 100)) +
# ylab("Fraction cells in C-D-E phase (%)")
# ggsave("Paryani_Accumbens_CELLTYPE_vs_fraction_SPN_CDE.pdf", width = 8, h
eight = 8)

#plot fraction of SPNs in C-D-E phase by HD grade
ggplot(uni_Paryani_Accumbens_Grade_CDE, aes(x = GRADE, y = Fract_CDE, color
= CELL_TYPE)) +
geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(w
idth = 0.1, height = 0)) +
scale_color_manual(values =c("#F8766D", "#7CAE00", "#00BFC4", "#C77CFF")) +
theme_classic() +
xlab("HD GRADE") +
ylim(c(0, 100)) +
ylab("Fraction cells in C-D-E phase (%)")
ggsave("Paryani_Accumbens_GRADE_vs_fraction_SPN_CDE.pdf", width = 6, height
= 6)
```

In [ ]:
```
# ggplot(uni_Paryani_Accumbens_Grade_CDE[which(uni_Paryani_Accumbens_Grade_
CDE$NUM_SPN > 30), ], aes(x = GRADE, y = Fract_CDE, color = CELL_TYPE)) +
# geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter
(width = 0.1, height = 0)) +
# scale_color_manual(values =c("#F8766D", "#7CAE00", "#00BFC4", "#C77CFF"))
+
# theme_classic() +
# xlab("HD GRADE") +
# ylim(c(0, 100)) +
# ylab("Fraction cells in C-D-E phase (%)")
# ggsave("Paryani_Accumbens_GRADE_vs_fraction_SPN_CDE_gt30.pdf", width = 6,
height = 6)
```

Plot_controls

In [ ]:
```
Handsaker_fract_CDE_tmp_Caudate_CTRL <- Handsaker_donor_metadata_CTRL
Handsaker_fract_CDE_tmp_Caudate_CTRL$SAMPLE <- rownames(Handsaker_donor_met
adata_CTRL)
Handsaker_fract_CDE_tmp_Caudate_CTRL$DATASET <- "Handsaker"
Handsaker_fract_CDE_tmp_Caudate_CTRL$TISSUE = "Caudate"
Handsaker_fract_CDE_tmp_Caudate_CTRL <- Handsaker_fract_CDE_tmp_Caudate_CTR
L[, c("DATASET", "SAMPLE", "PRED_FRACT_CDE", "NUM_SPN", "TISSUE")]
Handsaker_fract_CDE_tmp_Caudate_CTRL$Fract_SPN <- num_cells_Handsaker["S045
77", "num_spns"]/num_cells_Handsaker["S04577", "num_cells"]
```

In [ ]:
```
Lee_fract_CDE_tmp_Caudate_CTRL <- lapply(split(Lee_CTRL_MSN_Caudate@meta.da
ta, Lee_CTRL_MSN_Caudate@meta.data$NBB_ID), function(x) {
    SAMPLE=unique(x$NBB_ID)
    num_SPN <- dim(x)[1]
    num_CDE <- length(which(x$PREDICTED_PHASE == "C-D-E"))
    fraction_CDE <- num_CDE/num_SPN
    num_CAG <- gsub(x = x$CAG[1], pattern = "-.*", replacement = "")
    #cat(sprintf("Fraction MSNs in C-D-E phase: %.2f\n", fraction_CDE*100))
    df <- data.frame(DATASET = "Lee", SAMPLE, PRED_FRACT_CDE=fraction_CDE,
NUM_SPN = num_SPN, TISSUE = "Caudate")
    return(df)
})
#Lee_fract_CDE_tmp_Caudate_CTRL
Lee_fract_CDE_Caudate_CTRL <- do.call(rbind, Lee_fract_CDE_tmp_Caudate_CTR
L)
```

In [ ]:
```
Fract_SPN_Caudate <- c((num_cells_Lee["3345", "num_D1_SPN_caudate"] + num_c
ells_Lee["3345", "num_D2_SPN_caudate"])/num_cells_Lee["3345", "num_cells_ca
udate"],
                       (num_cells_Lee["4294", "num_D1_SPN_caudate"] + num_c
ells_Lee["4294", "num_D2_SPN_caudate"])/num_cells_Lee["4294", "num_cells_ca
udate"],
                       (num_cells_Lee["4308", "num_D1_SPN_caudate"] + num_c
ells_Lee["4308", "num_D2_SPN_caudate"])/num_cells_Lee["4308", "num_cells_ca
udate"],
                       (num_cells_Lee["4494", "num_D1_SPN_caudate"] + num_c
ells_Lee["4494", "num_D2_SPN_caudate"])/num_cells_Lee["4494", "num_cells_ca
udate"],
                       (num_cells_Lee["4621", "num_D1_SPN_caudate"] + num_c
ells_Lee["4621", "num_D2_SPN_caudate"])/num_cells_Lee["4621", "num_cells_ca
udate"],
                       (num_cells_Lee["A39R", "num_D1_SPN_caudate"] + num_c
ells_Lee["A39R", "num_D2_SPN_caudate"])/num_cells_Lee["A39R", "num_cells_ca
udate"],
                       (num_cells_Lee["A47L", "num_D1_SPN_caudate"] + num_c
ells_Lee["A47L", "num_D2_SPN_caudate"])/num_cells_Lee["A47L", "num_cells_ca
udate"])
Lee_fract_CDE_Caudate_CTRL$Fract_SPN <- Fract_SPN_Caudate
#Lee_fract_CDE_Caudate_CTRL
```

In [ ]:
```
Lee_fract_CDE_tmp_Putamen_CTRL <- lapply(split(Lee_CTRL_MSN_Putamen@meta.da
ta, Lee_CTRL_MSN_Putamen@meta.data$NBB_ID), function(x) {
    SAMPLE=unique(x$NBB_ID)
    num_SPN <- dim(x)[1]
    num_CDE <- length(which(x$PREDICTED_PHASE == "C-D-E"))
    fraction_CDE <- num_CDE/num_SPN
    num_CAG <- gsub(x = x$CAG[1], pattern = "-.*", replacement = "")
    #cat(sprintf("Fraction MSNs in C-D-E phase: %.2f\n", fraction_CDE*100))
    df <- data.frame(DATASET = "Lee", SAMPLE, PRED_FRACT_CDE=fraction_CDE,
NUM_SPN = num_SPN, TISSUE = "Putamen")
    return(df)
})
#Lee_fract_CDE_tmp_Caudate_CTRL
Lee_fract_CDE_Putamen_CTRL <- do.call(rbind, Lee_fract_CDE_tmp_Putamen_CTR
L)
```

In [ ]:
```
Fract_SPN_Putamen <- c((num_cells_Lee["4294", "num_D1_SPN_putamen"] + num_c
ells_Lee["4294", "num_D2_SPN_putamen"])/num_cells_Lee["4294", "num_cells_pu
tamen"],
                       (num_cells_Lee["4308", "num_D1_SPN_putamen"] + num_c
ells_Lee["4308", "num_D2_SPN_putamen"])/num_cells_Lee["4308", "num_cells_pu
tamen"],
                       (num_cells_Lee["4494", "num_D1_SPN_putamen"] + num_c
ells_Lee["4494", "num_D2_SPN_putamen"])/num_cells_Lee["4494", "num_cells_pu
tamen"],
                       (num_cells_Lee["A39R", "num_D1_SPN_putamen"] + num_c
ells_Lee["A39R", "num_D2_SPN_putamen"])/num_cells_Lee["A39R", "num_cells_pu
tamen"],
                       (num_cells_Lee["A47L", "num_D1_SPN_putamen"] + num_c
ells_Lee["A47L", "num_D2_SPN_putamen"])/num_cells_Lee["A47L", "num_cells_pu
tamen"])
Lee_fract_CDE_Putamen_CTRL$Fract_SPN <- Fract_SPN_Putamen
#Lee_fract_CDE_Putamen_CTRL
```

In [ ]:
```
Paryani_fract_CDE_tmp_Caudate_CTRL <- lapply(split(Paryani_CTRL_MSN_Caudate
@meta.data, Paryani_CTRL_MSN_Caudate@meta.data$Donor), function(x) {
    SAMPLE=unique(x$Donor)
    num_SPN <- dim(x)[1]
    num_CDE <- length(which(x$PREDICTED_PHASE == "C-D-E"))
    fraction_CDE <- num_CDE/num_SPN
    num_CAG <- gsub(x = x$CAG[1], pattern = "-.*", replacement = "")
    #cat(sprintf("Fraction MSNs in C-D-E phase: %.2f\n", fraction_CDE*100))
    df <- data.frame(DATASET = "Paryani", SAMPLE, PRED_FRACT_CDE=fraction_C
DE, NUM_SPN = num_SPN, TISSUE = "Caudate")
    return(df)
})
#Paryani_fract_CDE_tmp_Caudate_CTRL
Paryani_fract_CDE_Caudate_CTRL <- do.call(rbind, Paryani_fract_CDE_tmp_Caud
ate_CTRL)
#Paryani_fract_CDE_Caudate_CTRL
```

In [ ]:
```
Fract_SPN_Caudate <- c((num_cells_Paryani["T-4812", "num_dSPN1_caudate"] +
num_cells_Paryani["T-4812", "num_dSPN2_caudate"] + num_cells_Paryani["T-481
2", "num_iSPN1_caudate"] + num_cells_Paryani["T-4812", "num_iSPN2_caudat
e"])/num_cells_Paryani["T-4812", "num_cells_caudate"],
                       (num_cells_Paryani["T-4915", "num_dSPN1_caudate"] +
num_cells_Paryani["T-4915", "num_dSPN2_caudate"] + num_cells_Paryani["T-491
5", "num_iSPN1_caudate"] + num_cells_Paryani["T-4915", "num_iSPN2_caudat
e"])/num_cells_Paryani["T-4915", "num_cells_caudate"],
                       (num_cells_Paryani["T-5227", "num_dSPN1_caudate"] +
num_cells_Paryani["T-5227", "num_dSPN2_caudate"] + num_cells_Paryani["T-522
7", "num_iSPN1_caudate"] + num_cells_Paryani["T-5227", "num_iSPN2_caudat
e"])/num_cells_Paryani["T-5227", "num_cells_caudate"],
                       (num_cells_Paryani["T-5596", "num_dSPN1_caudate"] +
num_cells_Paryani["T-5596", "num_dSPN2_caudate"] + num_cells_Paryani["T-559
6", "num_iSPN1_caudate"] + num_cells_Paryani["T-5596", "num_iSPN2_caudat
e"])/num_cells_Paryani["T-5596", "num_cells_caudate"],
                       (num_cells_Paryani["T-5700", "num_dSPN1_caudate"] +
num_cells_Paryani["T-5700", "num_dSPN2_caudate"] + num_cells_Paryani["T-570
0", "num_iSPN1_caudate"] + num_cells_Paryani["T-5700", "num_iSPN2_caudat
e"])/num_cells_Paryani["T-5700", "num_cells_caudate"])

#Fract_SPN_Caudate
Paryani_fract_CDE_Caudate_CTRL$Fract_SPN <- Fract_SPN_Caudate
#Paryani_fract_CDE_Caudate_CTRL
```

In [ ]:
```
Paryani_fract_CDE_tmp_Accumbens_CTRL <- lapply(split(Paryani_CTRL_MSN_Accum
bens@meta.data, Paryani_CTRL_MSN_Accumbens@meta.data$Donor), function(x) {
    SAMPLE=unique(x$Donor)
    num_SPN <- dim(x)[1]
    num_CDE <- length(which(x$PREDICTED_PHASE == "C-D-E"))
    fraction_CDE <- num_CDE/num_SPN
    num_CAG <- gsub(x = x$CAG[1], pattern = "-.*", replacement = "")
    #cat(sprintf("Fraction MSNs in C-D-E phase: %.2f\n", fraction_CDE*100))
    df <- data.frame(DATASET = "Paryani", SAMPLE, PRED_FRACT_CDE=fraction_C
DE, NUM_SPN = num_SPN, TISSUE = "Accumbens")
    return(df)
})
#Paryani_fract_CDE_tmp_Accumbens_CTRL
Paryani_fract_CDE_Accumbens_CTRL <- do.call(rbind, Paryani_fract_CDE_tmp_Ac
cumbens_CTRL)
#Paryani_fract_CDE_Accumbens_CTRL
```

In [ ]:
```
Fract_SPN_Accumbens <- c((num_cells_Paryani["T-4915", "num_dSPN1_accumben
s"] +  num_cells_Paryani["T-4915", "num_dSPN2_accumbens"] + num_cells_Parya
ni["T-4915", "num_iSPN1_accumbens"] + num_cells_Paryani["T-4915", "num_iSPN
2_accumbens"])/num_cells_Paryani["T-4915", "num_cells_accumbens"],
                        (num_cells_Paryani["T-5227", "num_dSPN1_accumbens"]
+  num_cells_Paryani["T-5227", "num_dSPN2_accumbens"] + num_cells_Paryani
["T-5227", "num_iSPN1_accumbens"] + num_cells_Paryani["T-5227", "num_iSPN2_
accumbens"])/num_cells_Paryani["T-5227", "num_cells_accumbens"],
                        (num_cells_Paryani["T-5596", "num_dSPN1_accumbens"]
+  num_cells_Paryani["T-5596", "num_dSPN2_accumbens"] + num_cells_Paryani
["T-5596", "num_iSPN1_accumbens"] + num_cells_Paryani["T-5596", "num_iSPN2_
accumbens"])/num_cells_Paryani["T-5596", "num_cells_accumbens"],
                        (num_cells_Paryani["T-5700", "num_dSPN1_accumbens"]
+  num_cells_Paryani["T-5700", "num_dSPN2_accumbens"] + num_cells_Paryani
["T-5700", "num_iSPN1_accumbens"] + num_cells_Paryani["T-5700", "num_iSPN2_
accumbens"])/num_cells_Paryani["T-5700", "num_cells_accumbens"])

#Fract_SPN_accumbens
Paryani_fract_CDE_Accumbens_CTRL$Fract_SPN <- Fract_SPN_Accumbens
#Paryani_fract_CDE_Accumbens_CTRL
```

In [ ]:
```
CTRL_all_datasets <- rbind(Handsaker_fract_CDE_tmp_Caudate_CTRL, Lee_fract_
CDE_Caudate_CTRL, Lee_fract_CDE_Putamen_CTRL, Paryani_fract_CDE_Caudate_CTR
L, Paryani_fract_CDE_Accumbens_CTRL)
CTRL_all_datasets
```

In [ ]:

```
#plot results for CTRL donors
ggplot(CTRL_all_datasets, aes(x = TISSUE, color = DATASET, y = 100*PRED_FRA
CT_CDE, size = Fract_SPN)) +
geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(w
idth = 0.3, height = 0)) +
#scale_color_manual(values =c("#F8766D")) +
theme_classic() +
xlab("Tissue") +
ylim(c(0, 100)) +
ylab("Non-HD donors - Fraction cells in C-D-E phase (%)")
ggsave("CTRL_all_datasets_fraction_CDE.pdf", width = 8, height = 8)

ggplot(CTRL_all_datasets[which(CTRL_all_datasets$TISSUE == "Caudate"), ], a
es(x = TISSUE, color = DATASET, y = 100*PRED_FRACT_CDE, size = Fract_SPN))
+
geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(w
idth = 0.5, height = 0)) +
#scale_color_manual(values =c("#F8766D")) +
theme_classic() +
xlab("Tissue") +
ylim(c(0, 100)) +
ylab("Non-HD donors - Fraction cells in C-D-E phase (%)")
ggsave("CTRL_all_datasets_Caudate_fraction_CDE.pdf", width = 8, height = 8)

ggplot(CTRL_all_datasets[which(CTRL_all_datasets$TISSUE != "Caudate"), ], a
es(x = TISSUE, color = DATASET, y = 100*PRED_FRACT_CDE, size = Fract_SPN))
+
geom_point(aes(size = Fract_SPN), alpha = 0.8, position = position_jitter(w
idth = 0.3, height = 0)) +
#scale_color_manual(values =c("#F8766D")) +
theme_classic() +
xlab("Tissue") +
ylim(c(0, 100)) +
ylab("Non-HD donors - Fraction cells in C-D-E phase (%)")
ggsave("CTRL_all_datasets_nonCaudate_fraction_CDE.pdf", width = 8, height =
8)
#"#7CAE00", "#00BFC4", "#C77CFF"
```