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

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

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

# RNA seq analysis #

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

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

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

##load libraries

library(tximport)

library(AnnotationDbi)

library(tidyverse)

library(GenomicFeatures)

library(IRanges)

library(biomaRt)

library(DelayedArray)

library(DESeq2)

library(magrittr)

library(pheatmap)

library(RColorBrewer)

library(ggrepel)

library(apeglm)

library(gplots)

library(pcaMethods)

library(viridisLite)

library(ggplot2)

library(ConsensusClusterPlus)

library(NbClust)

library(factoextra)

getwd()

#load metadata

metadata <- read.csv("metadata.csv") #120

#can make new metadata tables if you want to look at particular conditions or else filter as below

#select for certain conditions

df <- filter(metadata, Timepoint == "T0") #60

df <- filter(metadata,

!sample\_id == "CC01\_T0",

!sample\_id == "CC11\_T0",

!sample\_id == "CC18\_T0",

!sample\_id == "CC24\_T0",

!sample\_id == "CC31\_T0") #55 (CCT0, SCT0), #43 (CCT0, SCT3)

df$cluster3 <- as.factor(df$cluster3)

# Make txdb object from GFF file

txdb <- makeTxDbFromGFF("gencode.v43.basic.annotation.gtf.gz",

format = "gtf")

saveDb(txdb, file="gencode.v43.sqlite")

#can just load this next time

txdb <- loadDb("gencode.v43.sqlite")

#select columns required

columns(txdb)

k <- keys(txdb, "GENEID")

tx2gene <- select(txdb, keys = k, keytype = 'GENEID', columns = 'TXNAME')

head(tx2gene)

res <- AnnotationDbi::select(txdb, k, "TXNAME", "GENEID")

tx2gene <- res[,2:1]

head(tx2gene)

#specify the working directory under the variable dir

dir <- "transcript\_level\_quantification"

#create files (edit the metadata to different analyses - eg, day 0 only)

list.files(file.path(dir))

files <- file.path(dir, dff$quant\_file)

names(files) <- dff$sample\_id #name files by sample name rather than quant file name

all(file.exists(files)) #should be true

#use tximport to read our count files. Since we are using the same gtf file, the versions of the transcripts will be the same, hence the argument ignoreTxVersion = FALSE. If not, we set it to TRUE

txi <- tximport(files = files, type = 'salmon', tx2gene = tx2gene, importer=read.delim,

ignoreTxVersion = FALSE, ignoreAfterBar = TRUE)

summary(txi)

head(txi)

#save the count file (abundance, counts, length)

write.csv(txi, "counts\_T0\_matrix.csv")

#save them individually (optional)

abundance <- txi$counts

write.csv(abundance, "counts\_T0\_matrix.csv")

levels(dds$cluster3)

dds$cluster3 <- relevel(dds$cluster3, ref = "2") #if required to relevel

# we now import the txi object into DESeq (~ tells DESEQ what to compare)

#design considers the effect of treatment/time etc

dds <- DESeqDataSetFromTximport(txi = txi,

colData = df,

design = ~library + cluster3) #wont take NA's

#whatever is first will be considered as a covariate, whatever is last is a factor for differential testing

#can add patient\_ID to adjust for that

#adjusting for batch effect

#include batch as a covariate in your design formula. By doing this, the effect of batch

#will be taken into account when your test statistics are derived.

#The regression model essentially treats it as a covariate and makes adjustment thus.

#This may not work, however, if the batch effect is large and inconsistent.

# scale the data for PCA (do this after running DeSeq2)

vsd <- vst(dds, blind=FALSE)

#code below is to look at the PCA when the batch isnt accounted for

pcaData\_raw <-plotPCA(vsd, intgroup=c("Condition", "library","case\_id"), returnData=TRUE)

percentVar\_raw <- round(100 \* attr(pcaData\_raw, "percentVar"))

label\_raw <- pcaData\_raw %>% filter(case\_id == "CC18" |

case\_id == "CC24" |

case\_id == "CC01" |

case\_id == "CC11" |

case\_id == "CC31"|

case\_id == "SC78")

PCA\_raw = ggplot(pcaData\_raw, aes(x = PC1, y =PC2))+

geom\_density\_2d(colour = "black", alpha = 0.2)+

geom\_point(show.legend = TRUE, aes(fill = Condition, shape = library), size =3) +

xlab(paste0("PC1: ",percentVar\_raw[1],"% variance")) +

ylab(paste0("PC2: ",percentVar\_raw[2],"% variance")) +

coord\_fixed()+

theme\_bw()+

#scale\_fill\_manual(values = c("1" = "navyblue",

# "2" = "#F7F0A6",

# "3" = "#D71B2B"))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d"))+

scale\_shape\_manual(values = c(21, 24))+

theme(legend.position = "right")+

#colours legend

guides(fill = guide\_legend(override.aes = list(shape=21)))+

xlim(-55,50)+

ylim(-50,50)+

geom\_label\_repel(data = label\_raw,

size = 2,

fill = "white",

min.segment.length = unit(0, 'lines'),

nudge\_y = 2,

aes(x=PC1,

y=PC2,

label=rownames(label\_raw)),

color="black")

PCA\_raw

ggsave(plot = PCA\_raw, "PCA\_Condition.png", width = 8, height = 6, dpi = 300)

ggsave(plot = PCA\_raw, "PCA\_vsd\_T0.png", width = 6, height = 5, dpi = 300)

#account for batch effect using limma (on vsd object which used dds post DeSeq2)

assay(vsd) <- limma::removeBatchEffect(assay(vsd), vsd$library)

#make data frame for vsd batch corrected count matrix por downstream analysis (making better PCA's/UMAPS)

vsd\_df <- as.data.frame(assay(vsd)) %>% rownames\_to\_column("GENEID")

vsd\_df$gene\_ID <- substr(vsd\_df$GENEID,1,15) #need gene\_id to align with gene name

colnames(vsd\_df)

ensembl = useMart( "ensembl", dataset = "hsapiens\_gene\_ensembl")

genemap <- getBM( attributes = c("ensembl\_gene\_id", "entrezgene\_id", "hgnc\_symbol"),

filters = "ensembl\_gene\_id",

values = vsd\_df$gene\_ID,

mart = ensembl)

idx <- match(vsd\_df$gene\_ID, genemap$ensembl\_gene\_id)

vsd\_df$entrez <- genemap$entrezgene[ idx ]

vsd\_df$hgnc\_symbol <- genemap$hgnc\_symbol[ idx ]

head(vsd\_df)

write\_csv(vsd\_df, "BSI/DisVHea\_vsd\_df\_limma.csv")

library(ggrepel)

#make a quick QC PCA to check data

pcaData <-plotPCA(vsd, intgroup=c("Condition", "library","case\_id"), returnData=TRUE, ntop = 12000)

percentVar <- round(100 \* attr(pcaData, "percentVar"))

label <- pcaData %>% filter(case\_id == "CC18" |

case\_id == "CC24" |

case\_id == "CC01" |

case\_id == "CC11" |

case\_id == "CC31"|

case\_id == "SC78")

batch\_corrected\_pca <- ggplot(pcaData, aes(PC1, PC2, color=Condition)) +

geom\_point(size=3) +

xlab(paste0("PC1: ",percentVar[1],"% variance")) +

ylab(paste0("PC2: ",percentVar[2],"% variance")) +

coord\_fixed()+

xlim(-50,50)

geom\_label\_repel(data = label\_raw,

size = 2,

fill = "white",

min.segment.length = unit(0, 'lines'),

nudge\_y = 2,

aes(x=PC1,

y=PC2,

label=rownames(label)),

color="black")

batch\_corrected\_pca

ggsave(plot = batch\_corrected\_pca, "PCA\_corrected.png", width = 8, height = 6, dpi = 300)

#save raw counts before normalising

dds\_df <- as.data.frame(counts(dds,normalized = FALSE)) %>% rownames\_to\_column("GENEID")

head(dds\_df)

dds\_df$gene\_ID <- substr(dds\_df$GENEID,1,15) #need gene\_id to align with gene name

genemap <- getBM( attributes = c("ensembl\_gene\_id", "entrezgene\_id", "hgnc\_symbol"),

filters = "ensembl\_gene\_id",

values = dds\_df$gene\_ID,

mart = ensembl )

idx <- match( dds\_df$gene\_ID, genemap$ensembl\_gene\_id )

dds\_df$entrez <- genemap$entrezgene[ idx ]

dds\_df$hgnc\_symbol <- genemap$hgnc\_symbol[ idx ]

head(dds\_df)

write\_csv(dds\_df, "Rawcounts\_dds\_all.csv")

# THIS IS A GOOD WAY TO MATCH GENE ID's to NAMEs and also entrez number.

library("biomaRt")

#the useEnsembl line worked as had to redirect to useast server. choose between useEnsembl and useMart option

#ensembl <- useEnsembl(biomart = "genes", dataset = "hsapiens\_gene\_ensembl", mirror = 'useast')

ensembl = useMart( "ensembl", dataset = "hsapiens\_gene\_ensembl")

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

# save the raw count data

dds\_df <- as.data.frame(counts(dds,normalized = TRUE)) %>% rownames\_to\_column("GENEID")

head(dds\_df)

dds\_df$gene\_ID <- substr(dds\_df$GENEID,1,15) #need gene\_id to allign with gene name

genemap <- getBM( attributes = c("ensembl\_gene\_id", "entrezgene\_id", "hgnc\_symbol"),

filters = "ensembl\_gene\_id",

values = dds\_df$gene\_ID,

mart = ensembl )

idx <- match( dds\_df$gene\_ID, genemap$ensembl\_gene\_id )

dds\_df$entrez <- genemap$entrezgene[ idx ]

dds\_df$hgnc\_symbol <- genemap$hgnc\_symbol[ idx ]

head(dds\_df)

write\_csv(dds\_df, "Normalisedcounts\_dds\_all.csv")

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

#before was just for PCA visualisation.. this is for removing fom data

dds\_df <- as.data.frame(limma::removeBatchEffect(counts(dds,normalized = FALSE), dds$library\_id)) %>% rownames\_to\_column("GENEID")

head(dds\_df)

dds\_df$gene\_ID <- substr(dds\_df$GENEID,1,15) #need gene\_id to allign with gene name

genemap <- getBM( attributes = c("ensembl\_gene\_id", "entrezgene\_id", "hgnc\_symbol"),

filters = "ensembl\_gene\_id",

values = dds\_df$gene\_ID,

mart = ensembl )

idx <- match( dds\_df$gene\_ID, genemap$ensembl\_gene\_id )

dds\_df$entrez <- genemap$entrezgene[ idx ]

dds\_df$hgnc\_symbol <- genemap$hgnc\_symbol[ idx ]

head(dds\_df)

write\_csv(dds\_df, "raw count data batch accounted.csv")

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

dds\_df\_norm <- as.data.frame(limma::removeBatchEffect(counts(dds, normalized = TRUE), dds$library\_id)) %>% rownames\_to\_column("GENEID")

dds\_df\_norm$gene\_ID <- substr(dds\_df\_norm$GENEID,1,15) #need gene\_id to allign with gene name

genemap <- getBM( attributes = c("ensembl\_gene\_id", "entrezgene\_id", "hgnc\_symbol"),

filters = "ensembl\_gene\_id",

values = dds\_df\_norm$gene\_ID,

mart = ensembl )

idx <- match( dds\_df\_norm$gene\_ID, genemap$ensembl\_gene\_id )

dds\_df\_norm$entrez <- genemap$entrezgene[ idx ]

dds\_df\_norm$hgnc\_symbol <- genemap$hgnc\_symbol[ idx ]

head(dds\_df\_norm)write\_csv(dds\_df\_norm, "normalized count data batch accounted.csv")

levels(dds$Condition) #levels are correct but if they werent...

levels(dds$Cohort)

dds$Condition <- relevel(dds$Condition, ref = "Healthy") #change to different tiimepoints for other comparisons

dim(dds)

#remove final number from ensembl name so it matches with annotation file

#ie change transcript to gene

head(rownames(dds))

table(duplicated(substr(rownames(dds),1,15)))

rownames(dds) <- make.unique(substr(rownames(dds),1,15))

head(rownames(dds))

#this is a filtering step - removes any counts with less than 10 counts in 5 samples

nrow(dds) #62263

ddsTxi <- dds[rowSums(counts(dds) >= 10) >= 5]

nrow(ddsTxi) #20838 (T0)

#run deseqobject

dds <- DESeq(ddsTxi)

design(dds)

resultsNames(dds)

res <- results(dds)

#can only do one comparison at a time

#use resultsNames(dds) from above

res <- results(dds, contrast = c("Cohort","Cardiac","Sepsis"),name= "Cohort\_Sepsis\_vs\_Cardiac")

res <- results(dds, contrast = c("Timepoint","T0","T1"),name= "Timepoint\_T1\_vs\_T0")

res <- results(dds, contrast = c("Timepoint","T2","T1"),name= "Timepoint\_T2\_vs\_T1")

res <- results(dds, contrast = c("Timepoint","T3","T1"),name= "Timepoint\_T3\_vs\_T1")

res <- results(dds, contrast = c("Timepoint","T3","T2"),name= "Timepoint\_T3\_vs\_T2")

res <- results(dds, contrast = c("Condition","Admission", "Healthy"),name= "Condition\_Admission\_vs\_Healthy")

res <- results(dds, contrast = c("cluster3","3","2"),name= "cluster3\_3\_vs\_2")

res.sort <- res[order(res$pvalue),]

res.sort

summary(res)

head(res)

res

plotMA(res, ylim=c(-1,1))

write.csv(res, "results\_admission\_vs\_healthy.csv")

#load annotation file

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

results <- read.csv("results\_admission\_vs\_healthy.csv")

#join annotation file to ensembl id for significant DEGs

DEG\_annot <- inner\_join(annotation, results, by = 'gene\_ID')

summary(DEG\_annot)

sum(is.na(DEG\_annot$gene\_name))

#replace blanks in gene\_name with NA

DEG\_annot[DEG\_annot == ""] <- NA

#get ensemble ID for NA genes

DEG\_NA <- DEG\_annot %>%

dplyr::filter(is.na (gene\_name)) %>%

dplyr::select(-gene\_name) %>%

dplyr::rename(gene\_name = 'gene\_ID')

#join ensemble ID NAs with gene names and remove NAs from gene name

DEG\_annot <- full\_join(DEG\_NA, DEG\_annot) %>%

dplyr::filter(!is.na(gene\_name))

head(DEG\_annot)

#removing 8th column (which is empty gene list)

DEG\_annot <- DEG\_annot[, -8]

write.csv(DEG\_annot, "results\_admission\_vs\_healthy\_annotated.csv")

# select genes that are statistically significant FDR<0.05

DEG\_p <- DEG\_annot %>%

dplyr::filter(padj <0.05, log2FoldChange < 1) %>%

dplyr::select(gene\_name, log2FoldChange) #1641 genes

summary(DEG\_p)

###### Volcano Plot ########

#load df

df <- read.csv("results\_admission\_vs\_healthy\_annotated.csv")

## add a column of NAs

df$diffexpressed <- "NO"

# if log2Foldchange > 0.6 and pvalue < 0.05, set as "UP"

df$diffexpressed[df$log2FoldChange > 1 & df$padj < 0.05] <- "UP"

# if log2Foldchange < -0.6 and pvalue < 0.05, set as "DOWN"

df$diffexpressed[df$log2FoldChange < -1 & df$padj < 0.05] <- "DOWN"

# Create a new column "delabel" to de, that will contain the name of genes differentially expressed (NA in case they are not)

df$delabel <- NA

df$delabel[df$diffexpressed != "NO"] <- df$gene\_name[df$diffexpressed != "NO"]

mycolors <- c("dodgerblue", "firebrick1", "grey")

names(mycolors) <- c("DOWN", "UP", "NO")

VP\_df <- ggplot(data=df, aes(x=(log2FoldChange), y=-log10(padj), label=delabel))+

geom\_point(aes(fill = diffexpressed), pch = 21, colour = "black", size = 2)+

theme\_classic()+

geom\_vline(xintercept=c(-1, 1), linetype="dashed", col="black") +

geom\_hline(yintercept=-log10(0.05), linetype="dashed", col="black")+

ggtitle("Day 9 vs Day 1")+

theme(plot.title = element\_text(hjust = 0.5))+

scale\_fill\_manual(values=mycolors)+

geom\_label\_repel(size = 2, force = 2, max.overlaps = 20)

xlim(c(-8,8))

ylim(c(0,10))

VP\_df

summary(df)

ggsave(plot = VP\_df, "remapintensive/VP\_CTS1\_seropositive\_D9v1.png", dpi = 300, height = 6, width = 6, units = "in")

MA\_df <- ggplot(data = df, aes(baseMean, log2FoldChange, label=delabel))+

geom\_point(aes(fill = diffexpressed), pch = 21, colour = "black", size = 1)+

scale\_x\_continuous(trans = "log10")+

scale\_fill\_manual(values=mycolors)+

geom\_hline(yintercept=0, col="black")+

geom\_hline(yintercept=1, col="black")+

geom\_hline(yintercept=-1, col="black")+

theme\_classic()+

geom\_label\_repel(size = 1.5, force = 2, max.overlaps = 40)

ylim(c(-5,5))

MA\_df

ggsave(plot = MA\_df, "MA\_SC\_T3vT0.png", dpi = 300, height = 6, width = 6, units = "in")

#GeneTonic for pathway analysis

#Load packages

library(GeneTonic)

library(topGO)

library(AnnotationDbi)

library(biomaRt)

library(org.Hs.eg.db)

library(pcaExplorer)

#Build Genetonic

#after you've run deseq!!!

#dds object above

design(dds)

levels(dds$Condition)

#take deseq object

#add symbols

rowData(dds)$SYMBOL1 <- get\_annotation\_orgdb(dds, "org.Hs.eg.db", "ENSEMBL")

rowData(dds)$SYMBOL <- as.factor(rowData(dds)$SYMBOL1$gene\_name)

library("AnnotationDbi")

# res object contrasting 2 conditions

#test against a null hypothesis of a log2FoldChange of 1 (instead of the default value of 0),

#in order to specify that we want to call DE genes with a consistent and robust expression change.

res\_CP <- results(dds,

contrast = c("Condition", "Discharge", "Healthy"),

lfcThreshold = 1, alpha = 0.05)

#add symbols

res\_CP$SYMBOL <- rowData(dds)$SYMBOL

head(res\_CP)

#we are going to extract the vector of DE genes (via deseqresult2df),

#as well as the list of genes to be used as background,

#feed these two objects into a function that computes the functional enrichment of the DE genes.

de\_symbols\_CP <- deseqresult2df(res\_CP, FDR = 0.05)$SYMBOL

bg\_ids <- rowData(dds)$SYMBOL[rowSums(counts(dds)) > 0]

head(de\_symbols\_CP)

#decorrelate the GO graph structure and deliver less redundant functional categories.

topgoDE\_day1\_group <-

pcaExplorer::topGOtable(de\_symbols\_CP,

bg\_ids,

ontology = "BP",

mapping = "org.Hs.eg.db",

geneID = "symbol",

topTablerows = 500)

#convert the output of the upstream tools into a format compatible with GeneTonic

res\_enrich\_day1group <- shake\_topGOtableResult(topgoDE\_day1\_group)

colnames(topgoDE\_day1\_group)

#> [1] "gs\_id" "gs\_description" "gs\_pvalue" "gs\_genes" "gs\_de\_count"

#> [6] "gs\_bg\_count" "Expected"

# annotation object

# i.e. a simple data frame composed at least of two columns, gene\_id,

#with a set of unambiguous identifiers (e.g. ENSEMBL ids)

#corresponding to the row names of the dds object, and gene\_name

anno\_df <- data.frame(

gene\_id = rownames(dds),

gene\_name = mapIds(org.Hs.eg.db,

keys = rownames(dds),

column = "SYMBOL",

keytype = "ENSEMBL"),

stringsAsFactors = FALSE,

row.names = rownames(dds)

)

#sometimes it is required to have aggregated scores for the res\_enrich data frame.

#This adds two columns to the provided res\_enrich object, z\_score and aggr\_score, which summarize at the gene set level the effect (log2FoldChange) of the differentially expressed genes which are its members

#In particular, the z score attempts to determine the “direction” of change,regardless of the effect size of the single members of the gene set.

res\_enrich\_day1group <- get\_aggrscores(res\_enrich = res\_enrich\_day1group,

res\_de = res\_CP,

annotation\_obj = anno\_df,

aggrfun = mean)

write.csv(res\_enrich\_day1group, "BSI/trajectory/GeneTonic\_DisVHealth.csv")

#All set!

GeneTonic(dds = dds,

res\_de = res\_CP,

res\_enrich = res\_enrich\_day1group,

annotation\_obj = anno\_df,

project\_id = "GT1")

#displaying the logFC of each set’s components -

#and transform the ggplot output into a plotly visualization,

#where tooltips can deliver additional information:

p <- enhance\_table(res\_enrich\_day1group,

res\_CP,

n\_gs = 30,

annotation\_obj = anno\_df,

chars\_limit = 60)

p

ggsave(plot = p, "BSI/trajectory/GeneTonic\_DisVHealth.png", dpi =300, width = 10, height = 6, units="in")

library(plotly)

ggplotly(p)

#

#

#

#alluvial

install.packages("ggalluvial")

library(ggalluvial)

alluvial <- read.csv("BSI/trajectory/3group.csv")

alluvial$cluster <- as.factor(alluvial$cluster)

alluvial$Condition <- factor(alluvial$Condition,

levels = c("Control", "Admission", "Day\_3", "Day\_5", "Discharge"), ordered = TRUE)

alluv <- ggplot(alluvial,

aes(x = Condition, stratum = cluster, alluvium = case\_id,

fill = cluster, label = cluster))+

scale\_fill\_manual(values=c("1" = "navyblue",

"2" = "#F7F0A6",

"3" = "#D71B2B"))+

#scale\_fill\_manual("cluster" = c("1" = "navyblue",

# "2" = "#F7F0A6",

# "3" = "#D71B2B"))+

geom\_flow(stat = "alluvium", lode.guidance = "frontback",

color = "darkgray") +

theme\_bw()+

geom\_stratum(aes(fill = cluster)) +

geom\_alluvium(aes(fill = cluster)) +

theme(legend.position = "right") +

ggtitle("Trajectory of clusters over time")+

theme(plot.title = element\_text(hjust = 0.5))+

xlab("Timepoint")+

ylab("Number of samples")

alluv

ggsave(plot = alluv, "BSI/trajectory/alluvial\_updated.png", dpi =300, width = 10, height = 8, units="in")

colnames(vsd\_df)

#heatmap with limma corrected vsd

df <- vsd\_df %>%

dplyr::select(gene\_ID, 2:116) %>% column\_to\_rownames("gene\_ID") %>% as.matrix()

colnames(df)

#T0 samples only

df <- vsd\_df %>%

dplyr::select(gene\_ID, 2:56) %>% column\_to\_rownames("gene\_ID") %>% as.matrix()

colnames(df)

#use consensus cluster plus package to select 5000 most variable genes (MARS group did this). this picks genes based on

# measured by median absolute deviation (mad)

library(ConsensusClusterPlus)

mads=apply(df,1,mad)

df1=df[rev(order(mads))[1:1000],]

#clustering

#use vsd object

d <- dist(df\_clust[2:115], method = "euclidean") # Euclidean distance matrix.

d <- dist(df3.1[10:1010], method = "euclidean") # Euclidean distance matrix.

H.fit <- hclust(d, method="ward")

abc = plot(H.fit) # display dendogram

groups <- cutree(H.fit, k=2) # cut tree into 5 clusters

# draw dendogram with red borders around the 5 clusters

rect.hclust(H.fit, k=3, border="red")

group <- as.data.frame(groups)#extract grouping variable

#make a sample Id column from rownames

cluster <- group %>% rownames\_to\_column("sample\_id")

H.fit$dist.method

png("Dendrogram\_1000.png", # create PNG for the heat map

width = 5\*300, # 5 x 300 pixels

height = 5\*300,

res = 300, # 300 pixels per inch

pointsize = 8) # smaller font size

H.fit

dev.off()

write.csv(df3.1, "df3.1\_limma.csv")

write.csv(group, "BSI/trajectory/3group.csv")

cluster <- full\_join(df3.1, cluster, by = "sample\_id")

ggsave(plot = abc, "clusters.png", dpi = 300, height = 6, width = 8, units = "in")

#use dds file

df1[1:5,1:5]

df1 <- df1 %>% as.data.frame() %>% rownames\_to\_column("gene\_id")

#want to get gene names for genes

library("biomaRt")

#the useEnsembl line worked as had to redirect to useast server. choose between useEnsembl and useMart option

#ensembl <- useEnsembl(biomart = "genes", dataset = "hsapiens\_gene\_ensembl", mirror = 'useast')

ensembl = useMart( "ensembl", dataset = "hsapiens\_gene\_ensembl")

#mart = useMart("ensembl")

genemap <- getBM( attributes = c("ensembl\_gene\_id", "hgnc\_symbol"),

filters = "ensembl\_gene\_id",

values = df1$gene\_id,

mart = ensembl)

idx <- match(df1$gene\_id, genemap$ensembl\_gene\_id)

df1$hgnc\_symbol <- genemap$hgnc\_symbol[ idx ]

colnames(df1)

#need to make genes variables and patients observations

df2 = setNames(data.frame(t(df1[,-1])), df1[,1]) #transpose so obvs = patients, variables = genes

#now have a data frame with 1000 of the most variable genes in the dataset.

df2 <- df2 %>% rownames\_to\_column("sample\_id")

#load in meta data

metadata <- read\_csv("metadata\_SRS.csv") %>%

dplyr::select("Cohort", "sample\_id", "case\_id",

"Cohort\_time", "Timepoint", "Condition", "library\_id", "library", "SRSq", "group") %>% #choose the variables I need

#filter(Timepoint == "T0") %>% #filter for T0

#filter(COVID\_status == "Negative") %>% #remove covid patients

#filter(!library\_id == "Cardiac\_lib2")%>% #remove duplicate samples

#filter(!case\_id == "SC09", #removed as IVIG

# !case\_id == "SC84", #removed as insisted on it

# !case\_id == "SC27") %>% #removed as screen failure, sofa >2

as.data.frame()

#merge with metadata

df3 <- merge(df2, metadata, all.x = TRUE)

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

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

# run a PCA on just cardiac and sepsis T0 samples

df3.1 <- df3 #%>% filter(Cohort\_time == "Sepsis\_T0" |

#Cohort\_time == "Cardiac\_T0")

table(df3.1$Cohort\_time)

df3.1$Cohort\_time

# 1) PCA

#runPCA

library(sva)

pca <- prcomp(df3.1[2:1001], scale. = FALSE)

pca1 <- as.data.frame(pca$x)

pca1

df3 <- cbind(df3.1, pca1) #add the pca to the df

write.csv(df3, "BSI/trajectory/variable\_genes.csv")

#pick up from here to make the heatmap (after adding the clusters)

df3 <- read.csv("BSI/variable\_genes\_T0\_lib2.csv")

#this is the plot for whats driving things

biplot <- fviz\_pca\_biplot(pca, repel = TRUE,

select.var = list(contrib = 10), #select number of variables - top 10 contributing factors

col.var = "contrib", # Variables color

gradient.cols = c("grey", "blue", "red"), #gradient colour scheme

col.ind = "grey", # Individuals color

alpha.ind = 0.3,

label = "var",

title = "",

pointsize = 3,

labelsize = 5)+

theme\_bw()+

theme(aspect.ratio = 1)

xlab(paste0("PC1: ",percentVar[1],"% variance")) +

ylab(paste0("PC2: ",percentVar[2],"% variance"))

biplot

ggsave(plot=biplot, "RNA\_biplot.png", dpi =300, height = 6, width = 6, units = "in")

#run fiz plot to get variation

fviz\_eig(pca, addlabels = TRUE, ylim = c(0, 50))

PCA\_plot <- ggplot(df3, aes(x = PC1, y = PC2))+

geom\_density\_2d(colour = "black", alpha = 0.2)+

geom\_point(show.legend = TRUE, aes(fill = Condition, shape = library), alpha = 0.9, size = 4, stroke = 0.7)+

theme\_bw()+

#guides(colour = guide\_legend(override.aes = list(size=5)))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca", "Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d"))+

scale\_shape\_manual(values = c(21, 24))+

#theme(aspect.ratio = 1)+

#guides(fill=guide\_legend(title="Cohort"))+

ylab("PC2")+

xlab("PC1")+

#guides(fill=guide\_legend(override.aes=list(shape=21)))+

ylim(-50,50)+

xlim(-65,65)+

theme(legend.position = "bottom",

legend.box = "vertical",

legend.title=element\_text(size=10, face = "bold"),

legend.text=element\_text(size=10),

axis.title =element\_text(size=10, face = "bold"))

PCA\_plot

ggsave(plot = PCA\_plot, "BSI/trajectory/PCA\_density.png", dpi=300, height = 5, width = 5, units = "in")

# added in clustering

df3.1 <- read.csv("BSI/trajectory/variable\_genes.csv")

#df3 <- df\_patient\_groups %>% cbind.data.frame(normalize(df\_patient\_groups[1:26]), df\_patient\_groups[27:64])

df3.1.1 <- df3.1

df3.1.1[2:1001] <- scale(df3.1.1[2:1001])

colnames(df3.1.1)

rownames(df3.1.1) <- df3.1.1$sample\_id

df3.1.1 <- df3.1.1 %>% dplyr::select(2:1001) %>% as.matrix()

library(circlize)

#sort out colour scheme

col\_fun = colorRamp2(c(-3, 0, 3), c("blue", "white", "red"))

col\_fun(seq(-3, 3))

library(ComplexHeatmap)

#annotation for row labels

row\_anno <- anno\_mark(at = c(1:115), labels = rownames(df3.1), which = "row",

link\_width = unit(1, "mm"),

link\_height = unit(1.5, "mm"),

padding = unit(0.55, "mm"),

labels\_rot = 0,

extend = unit(0, "mm"),

labels\_gp = gpar(fontsize = 2))

#draw label annotations for column

anno = anno\_mark(at = c(2:1001), labels = colnames(df3.1[2:1001]), which = "column", side = "bottom",

link\_width = unit(0.05, "mm"),

link\_height = unit(0.05, "mm"),

labels\_rot = 90,

padding = unit(0, "mm"),

extend = unit(0, "mm"),

labels\_gp = gpar(fontsize = 1))

#make annotation rows (patients)

colnames(df3.1)

anno\_df <- df3.1 %>% dplyr::select(Condition, SRS, cluster3)

colnames(anno\_df)

ha = rowAnnotation(df = anno\_df,

col = list(SRS = c("SRS1" = "darkcyan",

"SRS2" = "#A6F7C8",#A6F7C8

"SRS3" = "darkmagenta"),

Condition = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d"),

cluster3 = c("1" = "navyblue",

"2" = "#F7F0A6",

"3" = "#D71B2B"),

cluster2 = c("1" = "hotpink",

"2" = "slategray2"),

gp = gpar(col = "black", lwd = 2),

which = "row"))

ha

draw(ha, 1:95)

library(RColorBrewer)

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

my\_palette <- colorRampPalette(bluered(n=999))

colors <- c(seq(-2,-0.11,length=450),seq(-0.1,0.1,length=100),seq(0.11,2,length=450))

#### draw heatmap and include clustering from above

hm <- Heatmap(df3.1.1, name = "Unknown",

col = col\_fun,

cluster\_columns = TRUE,

cluster\_rows = H.fit,

show\_heatmap\_legend = TRUE,

clustering\_distance\_rows = "euclidean",

clustering\_method\_rows = "ward",

#bottom\_annotation = columnAnnotation(mark=anno),

# width = ncol(df4)\*unit(6, "mm"),

# height = nrow(df4)\*unit(4, "mm"),

column\_title\_gp = gpar(fontsize = 3, fontface = "bold"),

border = TRUE,

#row\_km = 5,

#column\_km = 3,

show\_column\_names = FALSE,

heatmap\_legend\_param = list(title = "Scale"),

column\_names\_gp = grid::gpar(fontsize = 2),

row\_names\_gp = grid::gpar(fontsize = 1),

right\_annotation = ha,

row\_dend\_width = unit(20, "mm"),

column\_dend\_side = c("top", "bottom"),

column\_dend\_height = unit(10, "mm"))#+

#rowAnnotation(mark=row\_anno)

hm

png("BSI/trajectory/heatmap\_rowTRUE\_3clust.png",width=10,height=10,units="in",res=1000)

hm

dev.off()

####

###

##

#

#SRS groupings

# Install dependencies

if (!requireNamespace("BiocManager", quietly = TRUE))

install.packages("BiocManager")

BiocManager::install("data.table", force = TRUE)

BiocManager::install("edgeR", force = TRUE)

# Install SepstratifieR

install.packages("devtools")

devtools::install\_github("jknightlab/SepstratifieR")

library(batchelor)

library(MatrixGenerics)

library(SepstratifieR)

library(data.table)

library(edgeR)

library(tidyverse)

# i only run this to get the variables for the normalisation downstream after running the cpm.

#it is important to do it in this order. Get counts, calcuate CPM, batch correction.

ntd <- normTransform(dds)

# extract the raw counts. Important it is raw counts here

counts <- counts(dds, normalized = FALSE)

DGEcpm <- as.data.frame(cpm(counts, log = TRUE, prior.count = TRUE), keep.rownames = TRUE) %>%

as.matrix()

?cpm

DGEcpm <- limma::removeBatchEffect(DGEcpm, ntd$library)%>% as.data.frame() %>% rownames\_to\_column("gene\_id")

DGEcpm$gene\_id <- substr(DGEcpm$gene\_id,1,15) #need gene\_id to align with gene name

head(counts)

head(DGEcpm)

# Stratify patients

#select genes for extended srs phenotyping

geneid <- data.frame (gene\_id = c("ENSG00000144659",

"ENSG00000103423",

"ENSG00000135372",

"ENSG00000079134",

"ENSG00000135972",

"ENSG00000087157",

"ENSG00000165006",

"ENSG00000111667",

"ENSG00000182670",

"ENSG00000097033",

"ENSG00000165733",

"ENSG00000103264",

"ENSG00000152219",

"ENSG00000100814",

"ENSG00000127334",

"ENSG00000131355",

"ENSG00000137337",

"ENSG00000156414",

"ENSG00000115085"),

gene\_name = c("SLC25A38",

"DNAJA3",

"NAT10",

"THOC1",

"MRPS9",

"PGS1",

"UBAP1",

"USP5",

"TTC3",

"SH3GLB1",

"BMS1",

"FBXO31",

"ARL14EP",

"CCNB1IP1",

"DYRK2",

"ADGRE3",

"MDC1",

"TDRD9",

"ZAP70")

)

colnames(DGEcpm)

dds\_df\_norm\_srs <- DGEcpm %>% filter(gene\_id %in% geneid$gene\_id) %>% dplyr::select(gene\_id, 2:121)

dds\_df\_norm\_srs = setNames(data.frame(t(dds\_df\_norm\_srs[,-1])), dds\_df\_norm\_srs[,1])%>% na.omit()

class(dds\_df\_norm\_srs)

predictions <- stratifyPatients(dds\_df\_norm\_srs, gene\_set = "extended")

davenport\_SRS\_predictions <- stratifyPatients(dds\_df\_norm\_srs, k = 144, gene\_set = "davenport")

a <- plotAlignedSamples(predictions)

a

ggsave(plot=a, "SRSextended\_acute.png", dpi = 300, width = 7, height = 8, units = "in")

b <- plotAlignedSamples(davenport\_SRS\_predictions)

b

ggsave(plot=b, "SRSdavenport\_acute.png", dpi = 300, width = 7, height = 8, units = "in")

pred <- data.frame(group = predictions@SRS,

SRSq = predictions@SRSq) %>% rownames\_to\_column("sample\_id")

write\_csv(pred, "srs\_groupings\_extended.csv")

pred1 <- data.frame(group = davenport\_SRS\_predictions@SRS,

SRSq = davenport\_SRS\_predictions@SRSq) %>% rownames\_to\_column("sample\_id")# %>% filter(!str\_detect(sample\_id, 'SC100'))

write\_csv(pred1, "srs\_groupings\_davenport.csv")

merged <- full\_join(pred, metadata, by = "sample\_id")

pred$cohort <- substr(pred$sample\_id, 1,2)

pred$timepoint <- substr(pred$sample\_id, 6,7)

pred <- pred %>% unite(Cohort\_time, c(cohort, timepoint), sep = "\_", remove = FALSE) %>% as.data.frame()

c <- ggplot(pred, aes(x=group, y = SRSq))+

geom\_boxplot()+geom\_jitter()

ggsave(plot=c, "SRSvSRSq\_acute.png", dpi = 300, width = 7, height = 8, units = "in")

d <- ggplot(merged, aes(x=Condition, y = SRSq))+

geom\_boxplot()+geom\_jitter()

ggsave(plot=d, "SRSq\_condition\_acute.png", dpi = 300, width = 7, height = 8, units = "in")

pred1$cohort <- substr(pred1$sample\_id, 1,2)

pred1$timepoint <- substr(pred1$sample\_id, 6,7)

pred1$patient\_id <- substr(pred1$sample\_id, 1,4)

pred1 <- pred1 %>% unite(Cohort\_time, c(cohort, timepoint), sep = "\_", remove = FALSE) %>% as.data.frame()

ggplot(pred1, aes(x=group, y = SRSq))+

geom\_boxplot()+geom\_jitter()

ggplot(pred1, aes(x=Cohort\_time, y = SRSq))+

geom\_boxplot()+geom\_jitter()

install.packages("easyalluvial")

library(easyalluvial)

head(merged)

col\_vector = c("SRS1" = "#010089",

"SRS2" = "#810103",

"SRS3" = "#4d7faf")

pred1 <- merged %>% filter(Cohort == "Sepsis")

head(pred1)

alluvial\_long(pred1

, key = Timepoint

, value = group

, id = sample\_id

, fill\_by = 'value'

, col\_vector\_flow = col\_vector

, col\_vector\_value = col\_vector

)

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

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

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

# cytokine analysis #

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

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

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

library(tidyverse)

library(ggplot2)

library(ggbeeswarm)

library(readr)

library(limma)

library(ggrepel)

getwd()

df <- read.csv("cytokines\_acute.csv") #120 samples

#filter for various things

df\_norm <- filter(df, timepoint == "T0)

colnames(df\_norm)

#cytokines

df\_log <- df\_norm %>% mutate\_at(vars(3:30), log2) %>%

select(3:30, patient\_id, sample\_id, Cohort\_time, Cohort, Timepoint, Condition, Cluster) %>%

as.data.frame()

colnames(df\_norm)

colnames(df\_log)

analysis <- df\_log %>% select("sample\_id", 1:28)

colnames(analysis)

#make matrix of values samples columns and frequencies as rows

count\_comparison\_tmydf = setNames(data.frame(t(analysis[,-1])), analysis[,1])%>% as.matrix()

str(count\_comparison\_tmydf)

dim(count\_comparison\_tmydf)

f <- factor(df\_log$Condition, levels=c("Healthy", "Admission", "Day\_3", "Day\_5", "Discharge"))

f <- factor(df\_log$Cluster, levels=c("1", "2", "3"))

design <- model.matrix(~0+f)

dim(design)

colnames(design) <- c("Healthy", "Admission", "Day\_3", "Day\_5", "Discharge")

colnames(design) <- c("one", "two", "three")

fit <- lmFit(count\_comparison\_tmydf, design)

contrast.matrix <- makeContrasts(Day\_3-Admission,

levels=design)

fit2 <- contrasts.fit(fit, contrast.matrix)

fit2 <- eBayes(fit2)

topTable(fit2)

?eBayes

df\_stats <- topTable(fit2, coef=1, adjust="BH", number = "inf") %>% rownames\_to\_column("Cytokine")

write.csv(df\_stats, "Limma\_T3vT0.csv")

getwd()

#load df

df\_stats <- read.csv("cytokines/Limma\_T2vT0.csv")

# add a column of NAs

df\_stats$diffexpressed <- "NO"

# if log2Foldchange > 0.6 and pvalue < 0.05, set as "UP"

df\_stats$diffexpressed[df\_stats$logFC > 0.5 & df\_stats$adj.P.Val < 0.05] <- "UP"

# if log2Foldchange < -0.6 and pvalue < 0.05, set as "DOWN"

df\_stats$diffexpressed[df\_stats$logFC < -0.5 & df\_stats$adj.P.Val < 0.05] <- "DOWN"

# Create a new column "delabel" to de, that will contain the name of genes differentially expressed (NA in case they are not)

df\_stats$delabel <- NA

df\_stats$delabel[df\_stats$diffexpressed != "NO"] <- df\_stats$Cytokine[df\_stats$diffexpressed != "NO"]

mycolors <- c("dodgerblue", "firebrick1", "grey")

names(mycolors) <- c("DOWN", "UP", "NO")

VP <- ggplot(data=df\_stats, aes(x=logFC, y=-log10( adj.P.Val), label=delabel)) +

geom\_point(aes(fill = diffexpressed), pch = 21, colour = "black", size = 4)+

theme\_classic()+

ggtitle("Cluster 3 vs 2")+

theme(plot.title = element\_text(hjust = 0.5, face= "bold", size = 16))+

geom\_vline(xintercept=c(-0.5, 0.5), col="black") +

geom\_hline(yintercept=-log10(0.05), col="black")+

scale\_fill\_manual(values=mycolors)+

geom\_label\_repel(size = 3, force = 5, max.overlaps =20)

VP

ggsave(plot = VP, "BSI/trajectory/VP\_cytokines-cluster3v2.png", dpi = 300, height = 6, width =6, units = "in")

#

#

#

#plot the log fold change to healthy

#

#

#

# Normalise to the median value of healthy

colnames(df)

colnames(fold\_change)

colnames(df)

fold\_change <- df %>% dplyr::select(3:30, patient\_id, sample\_id, Timepoint, Cohort\_time,

Cohort, Condition, Cluster) %>%

gather("Protein", "value",1:28) %>%

group\_by(Protein) %>%

na.omit %>%

mutate\_at(c("value"),funs(./median(.[Cluster == "1"]))) %>% as.data.frame()

colnames(df)

means <- df %>% dplyr::select(3:30, Condition) %>%

group\_by(Condition) %>%

summarise\_at(1:28,funs(mean(., na.rm=TRUE))) %>% #becayse its grouped\_by you need to check col indexes

as.data.frame()

median <- df %>% dplyr::select(3:30, Condition) %>%

group\_by(Condition) %>%

summarise\_at(1:28,funs(median(., na.rm=TRUE))) %>% #becayse its grouped\_by you need to check col indexes

as.data.frame()

#log10 the median values

median\_log <- log10(median(2:29))

median\_log <- median %>% mutate\_at(vars(2:29), log2)

#making a radar plot

library(fmsb)

colnames(median\_log)

# Define the variable ranges: maximum and minimum

max\_min <- data.frame(

APRIL = c(-2, 14), BAFF = c(-2, 14), sCD40L = c(-2, 14),

IL.5 = c(-2, 14), IL.13 = c(-2, 14), IL.2 = c(-2, 14),

IL.6 = c(-2, 14), IL.9 = c(-2, 14), IL.10 = c(-2, 14),

IFNy = c(-2, 14), TNFa = c(-2, 14), IL.17A = c(-2, 14),

IL.17F = c(-2, 14), IL.4 = c(-2, 14), IL.22 = c(-2, 14),

IL.8 = c(-2, 14), CXCL10 = c(-2, 14), CCL11 = c(-2, 14),

CCL17 = c(-2, 14), CCL2 = c(-2, 14), RANTES = c(-2, 14),

CCL3 = c(-2, 14), CXCL9 = c(-2, 14), CXCL5 = c(-2, 14),

CCL20 = c(-2, 14), CXCL1 = c(-2, 14), CXCL11 = c(-2, 14), CCL4 = c(-2, 14)

)

rownames(max\_min) <- c("Max", "Min")

#column to rownames

rownames(median\_log) <- median\_log$Condition

#remove the column

median\_log = subset(median\_log, select = -c(Condition))

# Bind the variable ranges to the data

df1 <- rbind(max\_min, median\_log)

df1

student1\_data <- df1[c("Min", "Max", "Healthy", "Admission", "Day\_3", "Day\_5", "Discharge"), ]

radarchart(student1\_data)

#this function creates a pretty layout

create\_beautiful\_radarchart <- function(data, color = "#00AFBB",

vlabels = colnames(df1), vlcex = 0.7,

caxislabels = NULL, title = NULL, ...){

radarchart(

data, axistype = 1,

# Customize the polygon

pcol = color, plwd = 2, plty = 1,

# Customize the grid

cglcol = "grey", cglty = 1, cglwd = 0.8,

# Customize the axis

axislabcol = "grey",

# Variable labels

vlcex = vlcex, vlabels = vlabels,

caxislabels = caxislabels, title = title, ...

)

}

# Reduce plot margin using par()

op <- par(mar = c(1, 2, 2, 1))

create\_beautiful\_radarchart(student1\_data, caxislabels = c(-2, 0, 2, 4, 6, 8, 10, 12, 14))

par(op)

# Reduce plot margin using par()

op <- par(mar = c(1, 2, 2, 2))

# Create the radar charts

create\_beautiful\_radarchart(

data = student1\_data, caxislabels = c(-4, 0, 4, 8, 12, 16),

color = c("Healthy" = "#55a8ca",

"Admission" ="#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d"

)

)

# Add an horizontal legend

legend(

x = -1.65, y =-0.55, legend = rownames(student1\_data[-c(1,2),]), horiz = FALSE,

bty = "n", pch = 20 , col = c("Healthy" = "#55a8ca",

"Admission" ="#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d"

),

text.col = "black", cex = 0.7, pt.cex = 1

)

par(op)

#fold\_change <- cbind(fold\_change, df$RN)

fold\_change$Condition <- factor(fold\_change$Condition,

levels = c("Healthy", "Admission", "Day\_3", "Day\_5", "Discharge"),

ordered = TRUE)

fold\_change$Cluster <- factor(fold\_change$Cluster)

# plot Log2 fold change

foldchange <- ggplot(fold\_change, aes(x=reorder(Protein, +as.numeric(l2fc)),y=l2fc))+

geom\_boxplot(aes(fill = Cluster),outlier.shape = NA,alpha=0.8)+ #position = position\_dodge(1)

geom\_point(aes(fill = Cluster), shape = 21,alpha=1, position = position\_jitterdodge())+

scale\_fill\_manual(values= c("1" = "navyblue",

"2" = "#F7F0A6",

"3" = "#D71B2B"))+

#scale\_fill\_manual(values= c("Admission" = "#ede151",

# "Day\_3" ="#f28e49",

# "Day\_5" = "#a65d86",

# "Discharge" = "#3c398d",

# "Healthy" = "#55a8ca"))+

#geom\_point(aes(fill = Cluster))+

theme(axis.text.x = element\_text(angle = 90, hjust = 1,vjust=0.4,size=4))+

theme\_classic()+

theme(axis.text.x = element\_text(angle = 90, hjust = 1,vjust=0.4,size=9))+

#geom\_hline(yintercept=0,linetype="dashed")+

#facet\_grid(Protein~., scales = "free\_y", space = "free\_y")+

coord\_flip()+

theme(legend.position = "bottom")+

theme(axis.text.x = element\_text(size=14, colour = "black"),

axis.text.y = element\_text(size=14, colour = "black"))+

theme(axis.title.x = element\_text(size=14, face= "bold", colour = "black"))+

theme(axis.title.y = element\_text(size=14, face= "bold", colour = "black"))+

ylab("Log2 fold change")+

xlab("Protein")+

theme(legend.text=element\_text(size=14))+

theme(legend.title=element\_text(size=14))

scale\_y\_continuous(limits=c(-15,10)) #needed to put as.numeric in the x and continuous in the y

foldchange

ggsave(plot = foldchange, "BSI/trajectory/cytokines\_foldchange\_clusters.png", dpi = 300, height = 15, width = 9)

df$Condition <- factor(df$Condition,

levels = c("Healthy", "Admission", "Day\_3", "Day\_5", "Discharge"),

ordered = TRUE)

#boxplot

#make it on a loop

#set up columns to plot (for cyto)

colnames(df)

response = names(df)[3:30] #this is columns that will be on y axis. Here selecting columns 1-26 to plot

response = set\_names(response)

expl = names(df)[35] #this is what column will be on x axis. Select one column like timepoint/phenotype

expl = set\_names(expl)

#

#below is the function

boxplot\_fun = function(x, y) {

ggplot(df %>% filter(!Cluster=='NA'), aes(x = .data[[y]], y = .data[[x]]))+

geom\_boxplot(aes(fill = Condition),shape = 21,alpha=0.4, outlier.shape = NA)+ # to. Create boxplot (could use geom\_violin as an. Alternative)

geom\_jitter(aes(fill = Condition),shape = 21,alpha=1)+

scale\_shape\_manual(values = c(21)) +

scale\_fill\_manual(values= c("Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d",

"Healthy" = "#55a8ca"))+ #choose your fill colours

scale\_y\_log10(expand = c(0, 0.5), breaks = scales::trans\_breaks("log10", function(x) 10^x),

labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+ #make a log10 scale, choosing what the breaks should look like. Here I have it to show 10to the power of x.

#annotation\_logticks(sides = "l")+ #adds log ticks to side of graph

theme\_classic()+ #choose whatever theme you. like. Can customise your own theme

theme(legend.position = "right")+ #remove legend - change to bottom, left etc to choose the position

guides(fill = guide\_legend(override.aes=list(shape=21)))+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+ #blank x axis. Can ovbs do this yourself

theme(axis.title.x = element\_text(size=12, colour = "black"))+ #customise text of axis

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, hjust =1, vjust =1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

theme(aspect.ratio = 1)+ #this makes the aspect ratio change. I wanted. A square look so changed to 1.

#add\_logticks(side = 'l', data = data.frame(x= NA, Condition = 'Healthy')) # this runs function above make sure # #plasma changes to CPAssignment variable and the = "CP" or whateever the variable is you need the llogticks on the left.

facet\_wrap(~Cluster, ncol =3)

}

#run it

all\_plots = map(expl,

~map(response, boxplot\_fun, y = .x) )

#check how the first three graphs look

all\_plots$Condition[1:3]

#We now want to save the plot. #where to save, then the dimensions and the dpi is quality. Always300.

pdf(file = "BSI/trajectory/boxplot\_cytokines\_facetCLUSTER.pdf", height = 7, width = 7)

all\_plots

dev.off()

#

#

#

#PCAs

colnames(df)

df\_log <- df %>% mutate\_at(vars(3:30), log2) %>% select(3:30, sample\_id, Condition) %>% as.data.frame() %>%

na.omit()

#select just the cytokine columns

pca <- prcomp(df\_log1, scale. = TRUE)

pca1 <- as.data.frame(pca$x)

pca1

#bind first 4 PC's to data set

df\_log <- cbind(df\_log,pca1[1:4])

#plot PCA

PCA\_plot <- ggplot(df\_log, aes(x = PC1, y = PC2, colour = Condition)) +

geom\_point(aes(fill = Condition),pch = 21, alpha = 1, size = 4, colour = "black")+

theme\_bw()+

guides(colour = guide\_legend(override.aes = list(size=5)))+

scale\_fill\_manual(values = c("Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d",

"Healthy" = "#55a8ca"))+

scale\_colour\_manual(values = c("Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d",

"Healthy" = "#55a8ca"))+

theme(aspect.ratio = 1)+

guides(fill=guide\_legend(title="Condition"))+

theme(text = element\_text(size=16),

legend.text=element\_text(size=16))+

stat\_ellipse()

PCA\_plot

ggsave(plot= PCA\_plot, "PCA\_cytokines.png", width = 7, height = 7, units = "in", dpi = 300)

library(factoextra)

biplot <- fviz\_pca\_biplot(pca, repel = TRUE,

select.var = list(contrib = 12), #select number of variables - top 10 contributing factors

col.var = "contrib", # Variables color

gradient.cols = c("grey", "blue", "red"), #gradient colour scheme

col.ind = "grey", # Individuals color

alpha.ind = 0.3,

label = "var",

title = "",

pointsize = 3,labelsize = 8)+

theme\_bw()+

theme(aspect.ratio = 1)+

ylab("PC2 (15.9%)")+

xlab("PC1 (34%)")+

theme(text = element\_text(size=16),

legend.text=element\_text(size=16))

biplot

ggsave(plot= biplot, "cytokines\_Biplot.png", width = 7, height = 7, units = "in", dpi = 300)

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

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

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

#correlations

colnames(df)

df <- df[-33]

df <- df[-19]

df <- df[-6]

# lets make a correlation network

library(circlize)

library(corrplot)

library(RColorBrewer)

library("Hmisc")

colnames(cytokines)

#make plot for each condition

hyp <- df %>% filter(Condition == "Healthy")

colnames(hyp)

res2 <- rcorr(as.matrix(hyp[3:30]), type = "spearman")

res <-cor(as.matrix(hyp[3:30]), method = "spearman")

corrplot(res2$r, type="upper", order="hclust",

p.mat = res2$P, sig.level = 0.01, insig = "blank")

corrplot(res2$r, type="lower", order="hclust",

col=rev(COL2("RdYlBu")), tl.col="black",

p.mat = res2$P, sig.level = 0.05,insig = "blank"

)

png()

png("corrplot\_Healthy.png", width = 6, height = 6, units='in', res = 300)

corrplot(res2$r, type="lower", order="hclust",

col=rev(COL2("RdYlBu")), tl.col="black",

p.mat = res2$P, sig.level = 0.05,insig = "blank"

)

dev.off()

flattenCorrMatrix <- function(cormat, pmat) {

ut <- upper.tri(cormat)

data.frame(

row = rownames(cormat)[row(cormat)[ut]],

column = rownames(cormat)[col(cormat)[ut]],

cor =(cormat)[ut],

p = pmat[ut]

)

}

library(ComplexHeatmap)

library(rstatix)

library(Hmisc)

options(scipen = 999)

res2\_df <- flattenCorrMatrix(res2$r, res2$P)

res2\_df <- res2\_df %>% adjust\_pvalue(p.col = "p", output.col = "p.adjust", method = "bonferroni")

res2\_df1 <- res2\_df %>% mutate(p.adjust = ifelse(p.adjust == 0, 1, p.adjust))

res2\_df2 <- res2\_df1%>% mutate(cor = ifelse(p.adjust > 0.05, NA, cor))

res2\_df2 <- res2\_df2 %>% select(-p, -p.adjust) %>% spread(row, cor) %>% column\_to\_rownames("column") %>% as.matrix()

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

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

# Nucleosome/Histone stuff #

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

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

histone <- read.csv("nucleosome/Histone\_combined.csv")

library(ggsignif)

library(ggpubr)

#make a table of signficance to export if you like

signficance\_H3R8 <- compare\_means(H3R8\_ng\_ml ~ Condition, data = histone, method = "wilcox.test", paired = FALSE)

write.csv(signficance\_H3R8, "nucleosome/stats\_H3R8.csv")

signficance\_H3.1 <- compare\_means(H3\_1\_ng\_ml ~ Condition, data = histone, method = "wilcox.test", paired = FALSE)

write.csv(signficance\_H3.1, "nucleosome/stats\_H3.1.csv")

#make a list of comparisons to feed into ggplot

my\_comparisons <- list( c("1", "2"), c("1", "3"), c("2", "3")

)

my\_comparisons <- list( c("1", "3")

)

my\_comparisons <- list( c("Control", "Admission"), c("Control", "Day\_3"), c("Control", "Day\_5"), c("Control", "Discharge"),

# c("Admission", "Day\_3"),c("Admission", "Day\_5"),

c("Admission", "Discharge"))

my\_comparisons <- list(#c("Healthy", "Admission"),

# c("Healthy", "Day\_3"),

c("Healthy", "Day\_5")

# c("Healthy","Discharge")

#c("Admission", "Day\_3")

# c("Admission", "Day\_5"),

# c("Admission", "Discharge")

)

#order

histone$Condition <- factor(histone$Condition,

levels = c("Control","Admission",

"Day\_3" , "Day\_5" , "Discharge"),

ordered = TRUE)

cytof$Condition <- factor(cytof$Condition,

levels = c("Healthy","Admission",

"Day\_3" , "Day\_5" , "Discharge"),

ordered = TRUE)

#boxplot

#

ggplot <- ggplot(histone, aes(x =Condition, y = H3R8\_ng\_ml ))+ #base ggplot code. Tell it the data frame then in aesthetics (aes) you add x and y variables

geom\_boxplot(aes(fill = Condition),shape = 21,alpha=0.4, outlier.shape = NA)+ # to. Create boxplot (could use geom\_violin as an. Alternative)

geom\_jitter(aes(fill = Condition),shape = 21,alpha=1)+#add this for dots over the box

#geom\_quasirandom(show.legend = TRUE, size =2, pch=21,

#aes(fill = Condition), stroke = 0.3, alpha = 1)+ #This adds individual dots in nice way. I like adding a fill to colour variables and have black outline. Looks cool. Stroke = width of the black outline, pch = shape.

scale\_fill\_manual(values= c("Control" = "#55a8ca",

"Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d"))+ #choose your fill colours

scale\_y\_log10(expand = c(0, 0.5), breaks = scales::trans\_breaks("log10", function(x) 10^x),

labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+ #make a log10 scale, choosing what the breaks should look like. Here I have it to show 10to the power of x.

#annotation\_logticks(sides = "l")+ #adds log ticks to side of graph

theme\_classic()+ #choose whatever theme you. like. Can customise your own theme

theme(legend.position = "bottom")+ #remove legend - change to bottom, left etc to choose the position

guides(fill = guide\_legend(override.aes=list(shape=21)))+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

ylab("H3R8 ng/ml")+#blank x axis. Can ovbs do this yourself

ggtitle ("H3R8")+

theme(plot.title = element\_text(size=16, colour = "black", face= "bold", vjust =0.5, hjust =0.5))+

theme(axis.title.x = element\_text(size=12, colour = "black"))+ #customise text of axis

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, hjust =1, vjust =1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=10))+

theme(aspect.ratio = 1) #this makes the aspect ratio change. I wanted. A square look so changed to 1.

stat\_compare\_means(comparisons = my\_comparisons, label = "p.signif", method = "wilcox.test", paired = FALSE)

#geom\_signif(comparisons = list(c("Healthy", "Discharge")),map\_signif\_level = TRUE)

ggplot

ggsave(plot= ggplot, "nucleosome/legend.png", width = 7, height = 7, units = "in", dpi = 300)

# major cell proportions (from cyTOF)

cytof <- read.csv("cyTOF.csv")

colnames(cytof)

cytof$cluster <- as.factor(cytof$cluster)

#same for cytof data

ggplot <- ggplot(cytof# %>% filter(!Condition=='NA')

, aes(x =Condition, y = X..of.T.cells.in.intact.cells))+ #base ggplot code. Tell it the data frame then in aesthetics (aes) you add x and y variables

geom\_boxplot(aes(fill = Condition),shape = 21,alpha=0.2, outlier.shape = NA)+ # to. Create boxplot (could use geom\_violin as an. Alternative)

geom\_jitter(aes(fill = Condition),shape = 21,alpha=1, size =3)+#add this for dots over the box

#geom\_quasirandom(show.legend = TRUE, size =2, pch=21,

#aes(fill = Condition), stroke = 0.3, alpha = 1)+ #This adds individual dots in nice way. I like adding a fill to colour variables and have black outline. Looks cool. Stroke = width of the black outline, pch = shape.

scale\_fill\_manual(values= c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day\_3" ="#f28e49",

"Day\_5" = "#a65d86",

"Discharge" = "#3c398d"))+ #choose your fill colours

#scale\_fill\_manual(values= c("1" = "navyblue",

# "2" = "#F7F0A6",

# "3" = "#D71B2B"))+

#scale\_y\_log10(expand = c(0, 0.5), breaks = scales::trans\_breaks("log10", function(x) 10^x),

#labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+ #make a log10 scale, choosing what the breaks should look like. Here I have it to show 10to the power of x.

#annotation\_logticks(sides = "l")+ #adds log ticks to side of graph

theme\_classic()+ #choose whatever theme you. like. Can customise your own theme

theme(legend.position = "right")+ #remove legend - change to bottom, left etc to choose the position

guides(fill = guide\_legend(override.aes=list(shape=21)))+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

ylab("% of intact cells")+#blank x axis. Can ovbs do this yourself

ggtitle ("T cells")+

#ylim(c(0,100))+

theme(plot.title = element\_text(size=16, colour = "black", face= "bold", vjust =0.5, hjust =0.5))+

theme(axis.title.x = element\_text(size=12, colour = "black"))+ #customise text of axis

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, hjust =1, vjust =1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

theme(aspect.ratio = 1)#this makes the aspect ratio change. I wanted. A square look so changed to 1.

stat\_compare\_means(comparisons = my\_comparisons, label = "p.signif", method = "wilcox.test", paired = FALSE)

#geom\_signif(comparisons = list(c("Healthy", "Discharge")),map\_signif\_level = TRUE)

ggplot

ggsave(plot= ggplot, "BSI/CD-CD19-\_intact.png", width = 5, height = 5, units = "in", dpi = 300)

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

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

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

# Mass cytometry #

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

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

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

##### CD4 T CELL UNSUPERVISED ANLAYSIS #### (analysis will be the same for all cell populations)

# T cell gated events have been downloaded as scale transformed CD19p events.

# 1) import the data

# 2) run the QC

# 3) run FlowSOM

# 4) extract the frequencies and export

# 5) extract the Median marker intensities and export

# 6) export large dataframe of all events with flowSOM clustering in case of future use

# 7) downsample events by patient for data visulisation

# 8)

# load packages

library(flowCore)

library(FlowSOM)

library(SingleCellExperiment)

library(plyr)

library(dplyr)

library(ggplot2)

library(scales)

library(reshape2)

library(RColorBrewer)

library(destiny)

library(uwot)

library(slingshot)

library(cytofkit2)

library(ggrepel)

library(pheatmap)

library(cowplot)

library(heatmaply)

library(tidyverse)

library(ggbeeswarm)

library(wesanderson)

library(purrr)

library(phateR)

######## LOAD DATA #########

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

## Provide the directory of the fcs files

dirFCS = "CD4 T cell fcs files"

## Defining a function to read multiple fcs files from a directory 'dir' into a single data.frame:

# NB: The column in the output named 'fileName' tracks the original file where each cell came from.

# Optionally perform remapping of column 'CSPLR\_ST' holding cytosplore sample numbers to actual names:

read.flowdat <- function(dir,path\_CSPLR\_ST=""){

# Read:

filepaths <- list.files(path=dir,pattern = ".fcs", full.names=TRUE)

flowset <- read.flowSet(files=filepaths, transformation=FALSE, truncate\_max\_range = FALSE)

# Transform to data frame:

x <- as.data.frame(exprs(as(flowset,'flowFrame')),stringsAsFactors=FALSE)

# Map column 'Original' to filename (in this case holding clusters of HSNE):

filenames <- gsub("[.fcs]","",list.files(path=dir,pattern = ".fcs", full.names=FALSE))

names(filenames) <- sort(unique(x$Original))

x$fileName <- filenames[as.character(x$Original)]

# Remove column 'Original':

x <- x[,-which(colnames(x)=="Original")]

# Optionally remap Cytosplore sample tags to original filename:

if(file.exists(path\_CSPLR\_ST)){

# Read:

sampID <- gsub(".fcs","",basename(sapply(strsplit(readLines(path\_CSPLR\_ST),": "),function(x) x[1])))

names(sampID) <- sapply(strsplit(readLines(path\_CSPLR\_ST),": "),function(x) x[2])

x$sampleID <- sampID[as.character(x$CSPLR\_ST)]

}

return(x)

}

# run function above. can see get file name column for cell.

df <- read.flowdat(dir=dirFCS)

colnames(df)

head(df)

df <- rownames\_to\_column(df, var = "Cell\_id") #this is important for later combining dataframes by unique number

# change colnames

colnames(df) <- c( "Cell\_id" = "Cell\_id",

"Time" = "Time",

"Event\_length" = "Event\_Length",

"Y89Di" = "Y89Di",

"Rh103Di" = "Rh103Di",

"Cd111Di" = "Cd111Di",

"Cd112Di" = "Cd112Di",

"Cd113Di" = "Cd113Di",

"Cd114Di" = "Cd114Di",

"In115Di" = "In115Di",

"Sn120Di" = "Sn120Di",

"I127Di" = "I127Di",

"Xe131Di" = "Xe131Di" ,

"Cs133Di" = "Cs133Di",

"Ba138Di" = "Ba138Di",

"Ce140Di" = "Beads",

"Pr141Di" = "CCR6",

"Nd142Di" = "CD19",

"Nd143Di" = "CD127",

"Nd144Di" = "CD38",

"Nd145Di" = "PD1",

"Nd146Di" = "IgD",

"Sm147Di" = "CD11c",

"Nd148Di" = "CD16",

"Sm149Di" = "CCR4",

"Nd150Di" = "CD119",

"Eu151Di" = "CD123",

"Sm152Di" = "TCRgd",

"Eu153Di" = "CXCR5",

"Sm154Di" = "CD3",

"Gd155Di" = "CD45RA",

"Gd156Di" = "PDL1",

"Gd157Di" = "Gd157Di",

"Gd158Di" = "CD27",

"Tb159Di" = "159Tb",

"Gd160Di" = "CD28",

"Dy161Di" = "CTLA4",

"Dy162Di" = "CD66b",

"Dy163Di" = "CXCR3",

"Dy164Di" = "CD161",

"Ho165Di" = "CD45RO",

"Er166Di" = "CD24",

"Er167Di" = "CCR7",

"Er168Di" = "CD8",

"Tm169Di" = "CD25",

"Er170Di" = "CD116",

"Yb171Di" = "CD20",

"Yb172Di" = "IgM",

"Yb173Di" = "HLADR",

"Yb174Di" = "CD4",

"Lu175Di" = "CD14",

"Yb176Di" = "CD56",

"BCKG190Di" = "BCKG190Di",

"Ir191Di" = "DNA1",

"Ir193Di" = "DNA2",

"Pt194Di" = "Pt194Di",

"Pt195Di" = "Pt195Di",

"Pt198Di" = "Pt198Di",

"Pb208Di" = "Pb208Di",

"Bi209Di" = "Bi209Di",

"Center" = "Center",

"Offset" = "Offset",

"Width" = "Width",

"Residual" = "Residual",

"fileName" = "Filename")

######## TRANSFORM DATA #########

#need to transform data.

# Set co-factor

cofac <- 5

colnames(df)

# Make arcsinh transformed expression matrix

df\_trans <- cbind.data.frame(asinh(df[17:53]/cofac), df[1], df[65])

# Plot two-marker example of transformation effect

p1 <- ggplot(df%>% slice\_sample(n = 10000), aes(x = CD45RA, y = CCR7)) +

geom\_point(color = 'grey', size = 0.01, alpha = 0.3) +

geom\_density2d() +

ggtitle('CD45RA vs. CCR7') +

theme\_bw() +

theme(plot.title = element\_text(hjust = 0.5))

p1

p2 <- ggplot(df\_trans%>% slice\_sample(n = 10000), aes(x = CD45RA, y = CCR7)) +

geom\_point(color = 'grey', size = 0.01, alpha = 0.3) +

geom\_density2d() +

ggtitle('CD45RA vs. CCR7 (transformed counts)') +

theme\_bw() +

theme(plot.title = element\_text(hjust = 0.5))

p3 <- plot\_grid(p1, p2, align = 'v', scale = 0.9)

p3

ggsave(plot= p3, "Figures/SCRIPT 1/QC/CD45RA\_CCR7\_trans\_comparison.png", width = 10, height = 10, units = "in", dpi = 300)

#look at transformations across all markers in T cell analysis

#also include list of markers to be included in downstream dataframes (this includes the state markers)

CD4\_cell\_markers <- c("CCR6", "CD38",

"CXCR5", "CD45RA", "CD45RO", "CD27", "CXCR3",

"CCR7", "CD25", "CD127", "HLADR", "CCR4", "CD28", "CD56")

markers\_for\_df <- c("CCR6", "CD38",

"CXCR5", "CD45RA", "CD45RO", "CD27", "CXCR3",

"CCR7", "CD25", "CD127", "HLADR", "CCR4", "CD28", "CD56", "PDL1", "PD1", "CTLA4","CD3", "CD4")

colnames(df\_trans)

#melt table

df\_ds <- df %>% slice\_sample(n = 10000)

df\_trans\_ds <- df\_trans %>% slice\_sample(n = 10000)

melt\_df <- melt(df\_ds[,markers\_for\_df]); melt\_df\_trans <- melt(df\_trans\_ds[,markers\_for\_df])

p1 <- ggplot(melt\_df, aes(y = value, x = variable)) +

geom\_boxplot(outlier.size=0.05) +

ggtitle('Raw ion counts') +

theme\_bw() + xlab("") +

theme(plot.title = element\_text(hjust = 0.5)) +

theme(axis.text.x = element\_text(angle = 90, hjust = 0.5))

p2 <- ggplot(melt\_df\_trans, aes(y = value, x = variable)) +

geom\_boxplot(outlier.size=0.05) +

ggtitle('ArcSinh transformed counts') +

theme\_bw() + xlab("") +

theme(plot.title = element\_text(hjust = 0.5)) +

theme(axis.text.x = element\_text(angle = 90, hjust = 0.5))

p3 <- plot\_grid(p1, p2, align = 'v', ncol = 1)

p3

ggsave(plot= p3, "Figures/SCRIPT 1/QC/all\_markers\_trans.png", width = 13, height = 10, units = "in", dpi = 300)

#data is now transformed and ready for cleaning

####### DATA CLEAN UP ##########

#remove samples with less than 1000 events

# check frequency of each cluster per sample

counts <- as.data.frame(table(df\_trans$Filename))

# a list of counts to be removed

counts\_remove <- counts %>% filter(Freq < 1000)

counts\_remove

#save as csv for info

write.csv(counts\_remove, "csv output/script 1/counts\_remove.csv")

# a list of counts to keep

counts\_keep <- counts %>% filter(Freq > 1000)

counts\_keep1 <- as.vector(counts\_keep$Var1)

write.csv(counts\_keep, "csv output/script 1/counts\_keep.csv")

#filter data frame to remove cells with less than 5000 in samples.

#select columns for downstream analysis and data effciency

df\_trans <- df\_trans %>% filter(Filename %in% counts\_keep1)%>%

select(one\_of(markers\_for\_df), Filename, Cell\_id)

###########CLUSTER CELLS ###############################

#FLOWSOM

colnames(df\_trans)

# implementing the cytof\_culster function that takes my expression dataframe as opposed to fcs files

#https://bioconductor.riken.jp/packages/3.3/bioc/vignettes/cytofkit/inst/doc/cytofkit\_example.html

# got the idea from

#https://biosurf.org/cytof\_data\_scientist.html#422\_flowsom\_clustering

#10 clusters have been selected to identify all relevent populations. 15 was done previously but too many

clusters\_fs <- cytof\_cluster(xdata = df\_trans[CD4\_cell\_markers], method = "FlowSOM", FlowSOM\_k = 10, flowSeed = 567)

clusters\_fs

#below I can run flowsom using the fcs files. I havent transformed these files. as they do in the script i was following

#https://github.com/janinemelsen/Single-cell-analysis-flow-cytometry/blob/master/scripts/CSV\_to\_transformed\_normalized\_FCS\_git.R

# combine flowsom data from cytof\_cluster to df

df\_trans <- cbind(df\_trans, FlowSOM=clusters\_fs)

colnames(df\_trans)

#write data here

# Want to normalise T cell parameters to a scale of 0-1 to easily visualize different expressions across different populations

# this function subtracts the minimum and divides by the maximum of all observations.

#Preserves the shape of each variables distribution while make them easily comparable on same

#scale

colnames(df\_trans)

df\_trans\_norm <- cbind.data.frame(normalize(df\_trans[markers\_for\_df]),df\_trans[20:22])

colnames(df\_trans\_norm)

# change colnames to have scale\_ in to be able to merge with other data frame.

colnames(df\_trans\_norm)[1:19] <- paste("Scale", colnames(df\_trans\_norm)[1:19], sep = "\_")

colnames(df\_trans\_norm)

##### MERGE NORM AND ORIGINAL#######

#merge the data frame with the original so have scaled and non scaled data

df1 <- merge(df\_trans, df\_trans\_norm)

colnames(df1)

#### calcuate frequencies

df1$Filename

# check frequency of each cluster per sample

counts <- as.data.frame.matrix(table(df1$Filename,df1$FlowSOM))

counts

#calculate percentages of a certain sample present in cluster

counts\_percofsample = counts/rowSums(counts)\*100

counts\_percofsample$total <- rowSums(counts)

counts\_percofsample$sampleID <- row.names(counts\_percofsample)

counts\_percofsample <- melt(counts\_percofsample, id.vars=c('sampleID', 'total'), variable.name='Population', value.name='frequency')

####################QC STEP - Any proportion under a median of 0.1% removed#####################

#medians of population

low\_pop <- counts\_percofsample %>%

group\_by(Population) %>%

summarize\_if(is.numeric, list(median)) %>%

as.data.frame()

low\_pop

# FILTER HERE IF NEEDED

### populations with a median of <0.1% are to be removed.

#filter out populations

df1 <- df1 %>% filter(!FlowSOM %in% c("8"))

#check they have been removed

table(df1$FlowSOM)

#check if that has now dramatically altered number of cells in samples

counts2 <- as.data.frame(table(df1$Filename))

counts\_remove2 <- counts2 %>% filter(Freq < 1000)

counts\_remove2

#it doesnt reduce samples to have very few counts so we are good. to proceed

#SAVE FILE HERE FOR USE ELSE WHERE

# E.g = extract proportions or medians or if need to use for future pipelines

write.csv(df1, "csv output/script 1/full\_data\_CD4\_T\_cells\_9\_clusters.csv")

###################### extract medians for heatmap ##########################################

#define clusters

#######IF NEEDED LOAD FILE

df1 <- read\_csv("csv output/script 1/full\_data\_CD4\_T\_cells\_9\_clusters.csv")

# extract medians for a heatmap (Using scaled)

class(df1$FlowSOM)

data\_median\_norm <- df1 %>%

group\_by(FlowSOM) %>%

summarize\_if(is.numeric, list(median)) %>%

select(contains("Scale"),FlowSOM) %>%

as.data.frame() %>%

column\_to\_rownames("FlowSOM")

table(df$FlowSOM, df$MEM\_label)

head(data\_median\_norm)

heatmap <- pheatmap(data\_median\_norm, color= viridis(50, option = "A"), cellheight=15, cellwidth = 15)

ggsave(plot= heatmap, "Figures/SCRIPT 1/flowSOM heatmap/heatmap\_norm\_CD4\_20231005.png", width = 10, height = 10, units = "in", dpi = 300)

write.csv(data\_median\_norm, "csv output/script 1/flowSOM\_norm\_expression\_231005.csv")

###################### re calcuate proportions after clean up################################

#### calcuate frequencies

df1$Filename

# check frequency of each cluster per sample

counts <- as.data.frame.matrix(table(df1$Filename,df1$FlowSOM))

counts

#calculate percentages of a certain sample present in cluster

counts\_percofsample = counts/rowSums(counts)\*100

counts\_percofsample$total <- rowSums(counts)

counts\_percofsample$Sample\_id <- row.names(counts\_percofsample)

counts\_percofsample <- melt(counts\_percofsample, id.vars=c('Sample\_id', 'total'), variable.name='Population', value.name='frequency')

counts\_percofsample$Sample\_id<-gsub("\_CD4\_T\_ell","",as.character(counts\_percofsample$Sample\_id))

counts\_percofsample$Sample\_id<-gsub("-\_1\_","",as.character(counts\_percofsample$Sample\_id))

# this removes the bit of fcs file name i dont need

counts\_percofsample

freq <- ggplot(counts\_percofsample, aes(x=reorder(Population, frequency, FUN=median), y=frequency))+

geom\_boxplot(position='dodge2', outlier.shape=NA)+

geom\_jitter(aes(colour=Sample\_id), size=2)+

xlab('cluster')+

theme(legend.position = "none")

freq

ggsave(plot= freq, "Figures/SCRIPT 1/QC/freq\_population\_boxplot.png", width = 10, height = 7, units = "in", dpi = 300)

rm(clinical)

#import the clinical data for grouping

clinical <- read\_csv("Clinical data/Cones V clinical data\_new.csv")

colnames(counts\_percofsample)

colnames(clinical)

colnames(counts\_percofsample)[1] <- "Sample\_id"

colnames(clinical)[1] <- "Sample\_id"

#merge with frequency

counts\_percofsample <- merge(counts\_percofsample, clinical)

# quick check

p5 <- ggplot(counts\_percofsample, aes(x=Condition, y=frequency))+

geom\_boxplot()+

geom\_quasirandom()+

facet\_wrap(~Population, scales = "free" )+

theme(axis.text.x=element\_text(angle=40, vjust=.9, hjust=1))+

theme(aspect.ratio = 1)

p5

ggsave(plot= p5, "Figures/SCRIPT 1/QC/Boxplot\_condiditon.png", width = 10, height = 10, units = "in", dpi = 300)

#stat\_summary(fun.y = median, geom="line", group= "Cohort", color= "blue", size = 1.5)

## write csv

#spread the data so can plot each one individually

counts\_percofsample\_spread <- counts\_percofsample %>% spread(Population, frequency)

colnames(counts\_percofsample\_spread)

write.csv(counts\_percofsample\_spread, "csv output/script 1/CD4\_T\_CELL\_frequency\_wide\_9\_clusters\_20231005.csv")

##IF NEEDED LOAD FILE

counts\_percofsample\_spread <- read\_csv("csv output/script 1/CD4\_T\_CELL\_frequency\_wide\_9\_clusters\_20230601.csv")

#manual individual checks

colnames(counts\_percofsample\_spread)

ggplot(counts\_percofsample\_spread, aes(x=Condition, y=`2`))+

geom\_boxplot()+

geom\_jitter()

######### BOXPLOT PROPORTIONS ##############

# a loop for proportions

colnames(counts\_percofsample\_spread)

#variables

response = names(counts\_percofsample\_spread)[6:14] #this is columns that will be on y axis

response = set\_names(response)

expl = names(counts\_percofsample\_spread)[5] #this is what column will be on x axis

expl = set\_names(expl)

#set the order of condition groups starting with healthy, admission, day3, day5 and discharge

counts\_percofsample\_spread$Condition <- factor(counts\_percofsample\_spread$Condition, levels=c('Healthy', 'Admission', 'Day 3', 'Day 5', 'Discharge'))

boxplot\_fun = function(x, y) {

ggplot(counts\_percofsample\_spread, aes(x = .data[[y]], y = .data[[x]]))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#47C455",

"Admission" = "#AD1A1A",

"Day 3" = "#e87135",

"Day 5" = "#8732a8",

"Discharge" = "#1A57AD"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))

}

colnames(counts\_percofsample\_spread)

boxplot\_fun

all\_plots = map(expl,

~map(response, boxplot\_fun, y = .x) )

all\_plots$Condition[1:2]

pdf("Figures/SCRIPT 1/population boxplots/CD4T\_cell\_boxplots\_prop\_9clusters\_20231005.pdf")

all\_plots

dev.off()

######### stacked proportions #######

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

##IF NEEDED LOAD FILE

counts\_percofsample\_spread <- read\_csv("csv output/script 1/CD4\_T\_CELL\_frequency\_wide\_9\_clusters\_20230601.csv")

##### relabel cluster columns

colnames(counts\_percofsample\_spread)

counts\_percofsample\_spread <- counts\_percofsample\_spread[ -c(1) ]

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "1"] <- "EM 1"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "2"] <- "EM"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "3"] <- "EM 2"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "4"] <- "Treg memory"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "5"] <- "CM"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "6"] <- "Naive"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "7"] <- "Treg memory 2"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "8"] <- "Act Naive"

names(counts\_percofsample\_spread)[names(counts\_percofsample\_spread) == "10"] <- "DP Transitional"

## This bit reorganises the dataframe

fold\_change <- counts\_percofsample\_spread %>% dplyr::select(1:14, Sample\_id, Condition) %>%

gather("Cell\_Type", "value", 6:14) %>%

group\_by(Cell\_Type) %>%

na.omit #%>% #remove this next line for cell stuff because it logs the data and divides by the mean

#mutate\_at(c("value"),funs(./median(.[Timepoint == "D1"]))) %>% as.data.frame()

fold\_change

#fold\_change$l2fc<-log(fold\_change$value,2) #also remove this line as we don’t want to log proportion data

fold\_change$Condition<- factor(fold\_change$Condition, levels =c ("Healthy", "Admission", "Day 3", "Day 5", "Discharge"))

library(ggplot2)

# plot Log2 fold change or (raw) value

foldchange <- ggplot(fold\_change,aes(x=reorder(Cell\_Type, +as.numeric(value)),y=value))+

geom\_boxplot(aes(fill = Condition),outlier.shape = NA,alpha=0.8, position = position\_dodge(1))+

scale\_fill\_manual(values= c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#geom\_point(aes(fill = Cluster))+

#theme(axis.text.x = element\_text(angle = 45, hjust = 0.2,vjust=0.4,size=4))+

theme\_classic()+

theme(axis.text.x = element\_text(angle = 45, hjust = 1,vjust=1,size=9))+

#rotate\_x\_text(angle = 45, align = 1, valign = 0.25)+

geom\_hline(yintercept=0,linetype="dashed")+

geom\_point(aes(fill= Condition), shape= 21, position = position\_jitterdodge(dodge.width = 1), size= 1.5)+

#facet\_grid(Cell\_Type, scales = "", space = "free\_y", ncol = 3)+

#coord\_flip()+

theme(legend.position = "bottom")+

theme(axis.text.x = element\_text(size=10, colour = "black"),

axis.text.y = element\_text(size=10, colour = "black"))+

theme(axis.title.x = element\_text(size=14, face= "bold", colour = "black"))+

theme(axis.title.y = element\_text(size=14, face= "bold", colour = "black"))+

ylab("Proportion")+

xlab("Cell Type")+

theme(legend.text=element\_text(size=14))+

theme(legend.title=element\_text(size=14))+

#coord\_flip()

#ylim(0,80)

###Make sure these are labelled in the correct order!!!!

scale\_x\_discrete(labels=c("Act Naive", "EM 1", "DP Transitional", "Treg memory 1", "EM", "Treg memory", "EM 2", "CM", "Naive")) #needed to put as.numeric in the x and continuous in the y

foldchange

ggsave(plot= foldchange, "Figures/SCRIPT 1/population boxplots/stacked\_bargraph\_9clusters\_20230612.png", width = 7, height = 7, units = "in", dpi = 300)

###### MMI BOXPLOTS #########

# Extract MMI's

## Extract median intensity per population per patient ########

### IF NEEDED LOAD FILE

df1 <- read\_csv("csv output/script 1/full\_data\_CD4\_T\_cells\_9\_clusters.csv")

colnames(df1)

data\_median <- df1 %>%

select(-Cell\_id) %>%

group\_by(Filename, FlowSOM) %>%

summarize\_if(is.numeric, list(median))

colnames(data\_median)

write.csv(data\_median, "csv output/script 1/CD4\_T\_CELL\_median\_check\_231005.csv")

#sum up rows of the same sampleid to make boxplots by marker

grouped\_median <- data\_median %>% group\_by(Filename) %>%

summarise\_all(.funs = sum,na.rm=T)

colnames(grouped\_median)

#Clean up data

grouped\_median$Filename<-gsub("\_CD4\_T\_ell","",as.character(grouped\_median$Filename))# this removes the bit of fcs file name i dont need

grouped\_median$Filename<-gsub("-\_1\_","",as.character(grouped\_median$Filename))

colnames(grouped\_median)[1] <- "Sample\_id"

#add the clinical data to median

grouped\_median <- merge(clinical, grouped\_median)

colnames(grouped\_median)

colnames(clinical)

write.csv(grouped\_median, "csv output/script 1/CD4\_T\_CELL\_median\_intensity\_groupedclusters\_20231005.csv")

#set the order of condition groups starting with healthy, admission, day3, day5 and discharge

grouped\_median$Condition <- factor(grouped\_median$Condition, levels=c('Healthy', 'Admission', 'Day 3', 'Day 5', 'Discharge'))

#plot generic boxplots of MMI's

names(grouped\_median)

response = names(grouped\_median)[6:24] #this is columns that will be on y axis

response = set\_names(response)

expl = names(grouped\_median)[4] #this is what column will be on x axis

expl = set\_names(expl)

boxplot\_fun = function(x, y) {

ggplot(grouped\_median, aes(x = .data[[y]], y = .data[[x]]) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))#+

# theme(aspect.ratio = 5)

#stat\_summary(fun.y = median, geom="line", group= "Cohort", size = 2, color = "black")

}

all\_plots = map(expl,

~map(response, boxplot\_fun, y = .x) )

all\_plots$Condition[1:2]

pdf("Figures/SCRIPT 1/population boxplots/CD4T\_cell\_boxplots\_MMI\_markers\_20231005.pdf")

all\_plots

dev.off()

##### DATA REDUCTION ########

# 1) down sample for quicker visulisation

# 2) PCA

# 3) UMAP

# 4) PHATE

# 5) diffusion map

colnames(df1)

# computer cant hack that number of events. do some downsampling

set.seed(123)#SET SEED HERE FOR REPRODUCIBILITY

df1\_ds <- df1 %>% group\_by(Filename) %>% slice\_sample(n = 1000)

table(df1\_ds$Filename) #check its worked

# 1) PCA

pca <- prcomp(df1\_ds[CD4\_cell\_markers], scale. = TRUE)

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

pca

#bind first 4 PC's to data set

df1\_ds <- cbind(df1\_ds,pca[1:4])

colnames(df1\_ds)

#quick check

PCA\_plot <- ggplot(df1\_ds, aes(x = PC1, y = PC2, color = as.factor(FlowSOM))) +

geom\_point(alpha = 0.6, size = 0.01) +

theme\_bw()+

guides(colour = guide\_legend(override.aes = list(size=5)))+

theme(aspect.ratio = 1)

PCA\_plot

ggsave(plot= PCA\_plot, "Figures/SCRIPT 1/Dimension reduction QC/9clustering\_PCA.pdf", width = 7, height = 7, units = "in", dpi = 300)

#2) UMAP

#need to set custom config to UMAP to ensure reproducibility

set.seed(8979)

umap <- umap(df1\_ds[CD4\_cell\_markers], n\_neighbors = 30, min\_dist=0.001, verbose=TRUE)

umap<- as.data.frame(umap)

colnames(umap) <- c('umap\_1', 'umap\_2')

df1\_ds <- cbind(df1\_ds,umap)

colnames(df1\_ds)

#quick check

UMAP\_plot <- ggplot(df1\_ds, aes(x = umap\_1, y = umap\_2, color = as.factor(FlowSOM))) +

geom\_point(alpha = 0.6, size = 0.01) +

theme\_bw()+

guides(colour = guide\_legend(override.aes = list(size=5)))

UMAP\_plot

ggsave(plot= UMAP\_plot, "Figures/SCRIPT 1/Dimension reduction QC/9clustering\_UMAP.pdf", width = 7, height = 7, units = "in", dpi = 300)

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

## Having run dimension reduction save the CSV

write\_csv(df1\_ds, "csv output/Script 1/downsample1000\_data\_CD4\_T\_CELL\_9clusters\_20231005.csv")

# plot the UMAPS

library(tidyverse)

library(viridis)

library(ggrepel)

library(RColorBrewer)

library(ggbeeswarm)

library(ggplot2)

#open the file

df <- read.csv("csv output/script 1/downsample1000\_data\_CD4\_T\_CELL\_9clusters\_20231005.csv")

#fix File name to Sample\_id

colnames(df)[1] <- "Sample\_id"

df$Sample\_id<-gsub("\_CD4\_T\_ell","",as.character(df$Sample\_id))# this removes the bit of fcs file name i dont need

df$Sample\_id<-gsub("\_1\_","",as.character(df$Sample\_id))

# read in clinical info

clinical <- read\_csv("Clinical data/Cones V clinical data\_new.csv")

head(clinical)

head(df)

colnames(clinical)[1] <- "Sample\_id"

#merge with clinical INFO

df <- merge(clinical, df)

#slice samples for even more speed

#df1 <- df %>% group\_by(Filename) %>% slice\_sample(n = 100)

#labels for UMAP

df <- df %>%

mutate(Population = case\_when(FlowSOM == "1" ~ "1: Activated EM CD4",

FlowSOM == "2" ~ "2: EM CD4",

FlowSOM == "3" ~ "3: EM CD127+ CD4",

FlowSOM == "4" ~ "4: Treg memory CD4",

FlowSOM == "5" ~ "5: CM CD4",

FlowSOM == "6" ~ "6: Naive CD4",

FlowSOM == "7" ~ "7: Exhausted Treg memory CD4",

FlowSOM == "9" ~ "9: Th2 CD4",

FlowSOM == "10" ~ "10: Activated Naive CD4" ))

names(df)

#Labels I have kept this code here but have removed the labels as not going to correct area

label\_flowsom\_umap <- df %>% group\_by(FlowSOM) %>% select(umap\_1, umap\_2) %>% summarize\_all(median)

class(df$umap\_2)

# Define the number of colors you want

nb.cols <- 9

mycolors <- colorRampPalette(brewer.pal(9, "Paired"))(nb.cols)

#plot labels on UMAP and filter by cohort\_time

UMAP\_labels <- ggplot(df, aes(x=umap\_1, y=umap\_2, ))+

geom\_point(aes(fill= Population), colour="black",pch=21, size=1, stroke=0.3)+

theme\_bw()+

scale\_fill\_manual(values = mycolors)+

# scale\_color\_brewer(palette = "Paired")+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

guides(shape = guide\_legend(override.aes = list(size = 10)))+

geom\_label\_repel(aes(label=FlowSOM), data=label\_flowsom\_umap, label.size = 0.25, size = 2.5,

force\_pull = 2)+guides(colour=FALSE)+

xlim(c(-10,8))+

ylim(c(-8,8))

UMAP\_labels

?geom\_label\_repel

ggsave(plot= UMAP\_labels, "Figures/SCRIPT 3/UMAP\_Labels\_9cluster\_20231005\_1.png", width = 7, height = 7, units = "in", dpi = 300)

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

#Key markers intensity

#IgM

colnames(df)

df$Scale\_CCR6

Scale\_CCR6 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CCR6),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+ ggtitle("CCR6")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CCR6

ggsave(plot= Scale\_CCR6, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CCR6.png", width = 5, height = 5, units = "in", dpi = 300)

Scale\_CD38 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD38),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1)) +

ggtitle("CD38")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD38

ggsave(plot= Scale\_CD38, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD38.png", width = 5, height = 5, units = "in", dpi = 300)

colnames(df)

#Scale\_CXCR5

Scale\_CXCR5 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CXCR5),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CXCR5")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CXCR5

ggsave(plot= Scale\_CXCR5, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CXCR5.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD45RA

Scale\_CD45RA <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD45RA),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD45RA")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD45RA

ggsave(plot= Scale\_CD45RA, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD45RA.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD45RO

Scale\_CD45RO <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD45RO),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD45RO")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD45RO

ggsave(plot= Scale\_CD45RO, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD45RO.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD27

Scale\_CD27 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD27),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD27")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD27

ggsave(plot= Scale\_CD27, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD27.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CXCR3

Scale\_CXCR3 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CXCR3),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CXCR3")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CXCR3

ggsave(plot= Scale\_CXCR3, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CXCR3.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CCR7

Scale\_CCR7 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CCR7),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CCR7")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CCR7

ggsave(plot= Scale\_CCR7, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CCR7.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD25

Scale\_CD25 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD25),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD25")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD25

ggsave(plot= Scale\_CD25, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD25.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD127

Scale\_CD127 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD127),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD127")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD127

ggsave(plot= Scale\_CD127, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD127.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_HLADR

Scale\_HLADR <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_HLADR),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("HLADR")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_HLADR

ggsave(plot= Scale\_HLADR, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_HLADR.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CCR4

Scale\_CCR4 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CCR4),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CCR4")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CCR4

ggsave(plot= Scale\_CCR4, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CCR4.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD28

Scale\_CD28 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD28),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD28")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD28

ggsave(plot= Scale\_CD28, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD28.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD56

Scale\_CD56 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD56),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD56")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD56

ggsave(plot= Scale\_CD56, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD56.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_PDL1

Scale\_PDL1 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_PDL1),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("PDL1")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_PDL1

ggsave(plot= Scale\_PDL1, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_PDL1.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_PD1

Scale\_PD1 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_PD1),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("PD1")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_PD1

ggsave(plot= Scale\_PD1, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_PD1.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CTLA4

Scale\_CTLA4 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CTLA4),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CTLA4")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CTLA4

ggsave(plot= Scale\_CTLA4, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CTLA4.png", width = 5, height = 5, units = "in", dpi = 300)

#Scale\_CD3

Scale\_CD3 <- ggplot(df, aes(x=umap\_1, y=umap\_2))+

geom\_point(aes(fill=Scale\_CD3),size=2, alpha = 1, pch = 21, colour = "black", stroke=0.2)+

theme\_bw()+

theme(aspect.ratio = 1) +

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

scale\_fill\_viridis(option = "magma", limits=c(0, 1))+

ggtitle("CD3")+

theme(plot.title = element\_text(vjust = -8, face = "bold",hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Scale\_CD3

ggsave(plot= Scale\_CD3, "Figures/SCRIPT 3/UMAP Intensitites/Scale\_CD3.png", width = 5, height = 5, units = "in", dpi = 300)

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

#PLOT THE DENSITY

# need to downsample so have equal number of cells in each group (Cohort\_Time)

table(df$Condition)#look at group with miniumum number of cells and use that as number for slice\_sample

set.seed(930)

density\_df <- df %>% group\_by(Condition) %>% slice\_sample(n = 6000)

table(density\_df$Condition)#check its worked

#Admission T0

Admission <- filter(density\_df, Condition == "Admission" ) #for density plot to overlay

Admission\_plot <- ggplot(density\_df, aes(x=umap\_1, y=umap\_2))+

geom\_point(colour ="grey", size=0.1, alpha = 1)+

geom\_point(data = Admission, colour ="black", size=0.1, alpha = 1)+

theme\_bw()+

theme(aspect.ratio = 1)+

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

#scale\_colour\_viridis(option = "A", limits=c(0, 1))+

#geom\_density\_2d\_filled(alpha = 0.9)+

geom\_density\_2d(size = 0.5, colour = "grey", contour\_var = "density", bins = 20, alpha = 0.5)+

geom\_density\_2d\_filled(data = Admission, aes(fill = after\_stat(level)),

contour\_var = "ndensity", # normalize to each QBs total passes

breaks = seq(0.1, 1.0, length.out = 20), alpha = 1)+ # drop the lowest passes)+

scale\_fill\_viridis(option="viridis",discrete=TRUE)+

ggtitle("Admission")+

theme(plot.title = element\_text(vjust = -6, face = "bold", hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Admission\_plot

ggsave(plot= Admission\_plot, "Figures/SCRIPT 3/Admission\_plot\_density\_contour20230523.png", width = 5, height = 5, units = "in", dpi = 300)

#Day3 T1

Day3 <- filter(density\_df, Condition == "Day 3" ) #for density plot to overlay

Day3\_plot <- ggplot(density\_df, aes(x=umap\_1, y=umap\_2))+

geom\_point(colour ="grey", size=0.1, alpha = 1)+

geom\_point(data = Day3, colour ="black", size=0.1, alpha = 1)+

theme\_bw()+

theme(aspect.ratio = 1)+

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

#scale\_colour\_viridis(option = "A", limits=c(0, 1))+

#geom\_density\_2d\_filled(alpha = 0.9)+

geom\_density\_2d(size = 0.5, colour = "grey", contour\_var = "density", bins = 20, alpha = 0.5)+

geom\_density\_2d\_filled(data = Day3, aes(fill = ..level..),

contour\_var = "ndensity", # normalize to each QBs total passes

breaks = seq(0.1, 1.0, length.out = 20), alpha = 1)+ # drop the lowest passes)+

scale\_fill\_viridis(option="viridis",discrete=TRUE)+

ggtitle("Day 3")+

theme(plot.title = element\_text(vjust = -6, face = "bold", hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Day3\_plot

ggsave(plot= Day3\_plot, "Figures/SCRIPT 3/Day3\_plot\_contour20230523.png", width = 5, height = 5, units = "in", dpi = 300)

#day5 T2

Day5 <- filter(density\_df, Condition == "Day 5" ) #for density plot to overlay

Day5\_plot <- ggplot(density\_df, aes(x=umap\_1, y=umap\_2))+

geom\_point(colour ="grey", size=0.1, alpha = 1)+

geom\_point(data = Day5, colour ="black", size=0.1, alpha = 1)+

theme\_bw()+

theme(aspect.ratio = 1)+

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

#scale\_colour\_viridis(option = "A", limits=c(0, 1))+

#geom\_density\_2d\_filled(alpha = 0.9)+

geom\_density\_2d(size = 0.5, colour = "grey", contour\_var = "density", bins = 20, alpha = 0.5)+

geom\_density\_2d\_filled(data = Day5, aes(fill = ..level..),

contour\_var = "ndensity", # normalize to each QBs total passes

breaks = seq(0.1, 1.0, length.out = 20), alpha = 1)+ # drop the lowest passes)+

scale\_fill\_viridis(option="viridis",discrete=TRUE)+

ggtitle("Day 5")+

theme(plot.title = element\_text(vjust = -6, face = "bold", hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Day5\_plot

ggsave(plot= Day5\_plot, "Figures/SCRIPT 3/Day5\_plot\_contour20230523.png", width = 5, height = 5, units = "in", dpi = 300)

#discharge T3

Discharge <- filter(density\_df, Condition == "Discharge" ) #for density plot to overlay

Discharge\_plot <- ggplot(density\_df, aes(x=umap\_1, y=umap\_2))+

geom\_point(colour ="grey", size=0.1, alpha = 1)+

geom\_point(data = Discharge, colour ="black", size=0.1, alpha = 1)+

theme\_bw()+

theme(aspect.ratio = 1)+

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

#scale\_colour\_viridis(option = "A", limits=c(0, 1))+

#geom\_density\_2d\_filled(alpha = 0.9)+

geom\_density\_2d(size = 0.5, colour = "grey", contour\_var = "density", bins = 20, alpha = 0.5)+

geom\_density\_2d\_filled(data = Discharge, aes(fill = ..level..),

contour\_var = "ndensity", # normalize to each QBs total passes

breaks = seq(0.1, 1.0, length.out = 20), alpha = 1)+ # drop the lowest passes)+

scale\_fill\_viridis(option="viridis",discrete=TRUE)+

ggtitle("Discharge")+

theme(plot.title = element\_text(vjust = -6, face = "bold", hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Discharge\_plot

ggsave(plot= Discharge\_plot, "Figures/SCRIPT 3/Discharge\_plot\_contour20230523.png", width = 5, height = 5, units = "in", dpi = 300)

#Healthy

Healthy <- filter(density\_df, Condition == "Healthy" ) #for density plot to overlay

Healthy\_plot <- ggplot(density\_df, aes(x=umap\_1, y=umap\_2))+

geom\_point(colour ="grey", size=0.1, alpha = 1)+

geom\_point(data = Healthy, colour ="black", size=0.1, alpha = 1)+

theme\_bw()+

theme(aspect.ratio = 1)+

theme(panel.grid.major = element\_blank(), panel.grid.minor=element\_blank())+

#scale\_colour\_viridis(option = "A", limits=c(0, 1))+

#geom\_density\_2d\_filled(alpha = 0.9)+

geom\_density\_2d(size = 0.5, colour = "grey", contour\_var = "density", bins = 20, alpha = 0.5)+

geom\_density\_2d\_filled(data = Healthy, aes(fill = ..level..),

contour\_var = "ndensity", # normalize to each QBs total passes

breaks = seq(0.1, 1.0, length.out = 20), alpha = 1)+ # drop the lowest passes)+

scale\_fill\_viridis(option="viridis",discrete=TRUE)+

ggtitle("Healthy")+

theme(plot.title = element\_text(vjust = -6, face = "bold", hjust = 0.05))+

theme(legend.position = "none")+

ylab("")+

xlab("")+

xlim(c(-10,10))+

ylim(c(-8,9))

Healthy\_plot

ggsave(plot= Healthy\_plot, "Figures/SCRIPT 3/Healthy\_plot\_density\_contour20230523.png", width = 5, height = 5, units = "in", dpi = 300)

#######################proportions ##########################

### plot the boxplots for proportions

df\_bp <- read\_csv("csv output/script 1/CD4\_T\_CELL\_frequency\_wide\_14clusters\_20230511.csv")

colnames(df\_bp)

df\_bp1 <- df\_bp %>% rename(`1: CD38+ HLA-DR+ Activated CD4` = `1`,

`2: TEMRA CD4` = `2`,

`3: Naive CD4` = `3`,

`4: CD4RO+ CD4` = `4`,

`5: ? CD4` = `5`,

`6: Th2 CD38+ CD4` = `6`,

`8: Tfh CD4` = `8`,

`9: Th2 CM CD38+ CD4` = `9`,

`10: Th1 EM CD4` = `10`,

`11: Th2 EM CD25+ CD4` = `11`,

`12: Th2 CM CCR4+ CD4` = `12`,

`13: Naive CD25+ CD4` = `13`,

`14: Treg CD4` = `14`,

`15: Treg memory CD4` = `15`,)

colnames(df\_bp1)

#rename columns with population names

response = names(df\_bp1)[7:20] #this is columns that will be on y axis

response = set\_names(response)

expl = names(df\_bp1)[6] #this is what column will be on x axis

expl = set\_names(expl)

boxplot\_fun = function(x, y) {

ggplot(df\_bp1, aes(x = .data[[y]], y = .data[[x]]))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Admission" = "#AD1A1A",

"Day 3" = "#f28888",

"Day 5" = "#eb9bcd",

"Discharge" = "#c9f2f2",

"Healthy" = "#47C455"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))

}

all\_plots = map(expl,

~map(response, boxplot\_fun, y = .x) )

all\_plots$Condition[1:2]

pdf("Figures/SCRIPT 3/CD4T\_cell\_boxplots\_prop\_14clusters\_20230523.pdf")

all\_plots

dev.off()

#### STATS on MMI Boxplots

library(ggplot2)

library(ggpubr)

library(plyr)

library(dplyr)

library(ggplot2)

library(scales)

library(reshape2)

library(RColorBrewer)

library(destiny)

library(uwot)

library(slingshot)

library(ggrepel)

library(cowplot)

library(tidyverse)

library(ggbeeswarm)

library(wesanderson)

library(purrr)

# Extract MMI's

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

## Extract median intensity per population per patient ########

### IF NEEDED LOAD FILE

df1 <- read\_csv("csv output/script 1/full\_data\_CD4\_T\_cells\_9\_clusters.csv")

df1 <- df1[ -c(1) ]

colnames(df1)

data\_median <- df1 %>%

select(-Cell\_id) %>%

group\_by(Filename, FlowSOM) %>%

summarize\_if(is.numeric, list(median))

colnames(data\_median)

#sum up rows of the same sampleid to make boxplots by marker

grouped\_median <- data\_median %>% group\_by(Filename) %>%

summarise\_all(.funs = sum,na.rm=T)

colnames(grouped\_median)

#Clean up data

grouped\_median$Filename<-gsub("\_CD4\_T\_ell","",as.character(grouped\_median$Filename))# this removes the bit of fcs file name i dont need

grouped\_median$Filename<-gsub("-\_1\_","",as.character(grouped\_median$Filename))

colnames(grouped\_median)[1] <- "Sample\_id"

#add the clinical data to median

# read in clinical info

clinical <- read\_csv("Clinical data/Cones V clinical data\_new.csv")

grouped\_median <- merge(clinical, grouped\_median)

colnames(grouped\_median)

colnames(clinical)

write.csv(grouped\_median, "csv output/script 1/CD8\_T\_CELL\_median\_intensity\_groupedclusters\_20230608.csv")

grouped\_median <- read\_csv("csv output/script 1/CD4\_T\_CELL\_median\_intensity\_groupedclusters\_20230607.csv")

grouped\_median <- grouped\_median[ -c(1) ] #removes the unwanted first column when loading dataframe

#set the order of condition groups starting with healthy, admission, day3, day5 and discharge

grouped\_median$Condition <- factor(grouped\_median$Condition, levels=c('Healthy', 'Admission', 'Day 3', 'Day 5', 'Discharge'))

#plot generic boxplots of MMI's

names(grouped\_median)

response = names(grouped\_median)[7:24] #this is columns that will be on y axis

response = set\_names(response)

expl = names(grouped\_median)[4] #this is what column will be on x axis

expl = set\_names(expl)

######CHECK FOR NORMALLY DISTRIBUTED VALUES

#if significant (<0.05) then they are not normal and wilcox test should be used. Otherwise t.test can be used

shapiro.test(grouped\_median$'HLADR'[grouped\_median$Condition=="Healthy"]) #normal

shapiro.test(grouped\_median$'HLADR'[grouped\_median$Condition=="Admission"]) #not normal

shapiro.test(grouped\_median$'HLADR'[grouped\_median$Condition=="Day 3"]) #not normal

shapiro.test(grouped\_median$'HLADR'[grouped\_median$Condition=="Day 5"]) #normal

shapiro.test(grouped\_median$'HLADR'[grouped\_median$Condition=="Discharge"]) #normal

shapiro.test(grouped\_median$'PD1'[grouped\_median$Condition=="Healthy"]) #normal

shapiro.test(grouped\_median$'PD1'[grouped\_median$Condition=="Admission"]) #normal

shapiro.test(grouped\_median$'PD1'[grouped\_median$Condition=="Day 3"]) #normal

shapiro.test(grouped\_median$'PD1'[grouped\_median$Condition=="Day 5"]) #normal

shapiro.test(grouped\_median$'PD1'[grouped\_median$Condition=="Discharge"]) #normal

shapiro.test(grouped\_median$'CD25'[grouped\_median$Condition=="Healthy"]) #normal

shapiro.test(grouped\_median$'CD25'[grouped\_median$Condition=="Admission"]) #normal

shapiro.test(grouped\_median$'CD25'[grouped\_median$Condition=="Day 3"]) #normal

shapiro.test(grouped\_median$'CD25'[grouped\_median$Condition=="Day 5"]) #normal

shapiro.test(grouped\_median$'CD25'[grouped\_median$Condition=="Discharge"]) #normal

shapiro.test(grouped\_median$'CD127'[grouped\_median$Condition=="Healthy"]) #normal

shapiro.test(grouped\_median$'CD127'[grouped\_median$Condition=="Admission"]) #normal

shapiro.test(grouped\_median$'CD127'[grouped\_median$Condition=="Day 3"]) #normal

shapiro.test(grouped\_median$'CD127'[grouped\_median$Condition=="Day 5"]) #normal

shapiro.test(grouped\_median$'CD127'[grouped\_median$Condition=="Discharge"]) #normal

shapiro.test(grouped\_median$'CXCR5'[grouped\_median$Condition=="Healthy"]) #normal

shapiro.test(grouped\_median$'CXCR5'[grouped\_median$Condition=="Admission"]) #not normal

shapiro.test(grouped\_median$'CXCR5'[grouped\_median$Condition=="Day 3"]) #not normal

shapiro.test(grouped\_median$'CXCR5'[grouped\_median$Condition=="Day 5"]) #not normal

shapiro.test(grouped\_median$'CXCR5'[grouped\_median$Condition=="Discharge"]) #not normal

##### Make a list to use as a comparison i.e Healthy vs other conditions.

healthy\_vs <- list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"))

#####Box plot with one marker to check

CCR6 <- ggplot(grouped\_median, aes(x = Condition, y = CCR6) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Day 3"), c("Healthy", "Day 5"),

c("Admission", "Day 3"), c("Admission", "Day 5"),

c("Day 3", "Discharge")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

#run boxplot

CCR6

ggsave(plot= CCR6, "Figures/SCRIPT 1/Population boxplots/CD8T\_cell\_boxplots\_MMI\_markers\_20230612\_statssig\_CCR6.pdf", width = 7, height = 7, units = "in", dpi = 300)

#####

HLADR <- ggplot(grouped\_median, aes(x = Condition, y = HLADR) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 5"),

c("Day 3", "Day 5")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

#run boxplot

HLADR

ggsave(plot= HLADR, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_MMI\_markers\_20231004\_statssig\_HLADR.pdf", width = 5, height = 5, units = "in", dpi = 300)

######

PD1 <- ggplot(grouped\_median, aes(x = Condition, y = PD1) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

#run boxplot

PD1

ggsave(plot= PD1, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_MMI\_markers\_20231004\_statssig\_PD1.pdf", width = 4, height = 4, units = "in", dpi = 300)

######

PDL1 <- ggplot(grouped\_median, aes(x = Condition, y = PDL1) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"),

c("Admission", "Day 5")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

#run boxplot

PDL1

ggsave(plot= PDL1, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_MMI\_markers\_20231004\_statssig\_PDL1.pdf", width = 5, height = 5, units = "in", dpi = 300)

######

CD25 <- ggplot(grouped\_median, aes(x = Condition, y = CD25) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Discharge"),

c("Day 3", "Day 5"),

c("Day 5", "Discharge")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

#run boxplot

CD25

ggsave(plot= CD25, "Figures/SCRIPT 1/Population boxplots/CD8T\_cell\_boxplots\_MMI\_markers\_20230914\_statssig\_CD25.pdf", width = 7, height = 7, units = "in", dpi = 300)

######

CD127 <- ggplot(grouped\_median, aes(x = Condition, y = CD127) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

#run boxplot

CD127

ggsave(plot= CD127, "Figures/SCRIPT 1/Population boxplots/CD8T\_cell\_boxplots\_MMI\_markers\_20230912\_statssig\_CD127.pdf", width = 7, height = 7, units = "in", dpi = 300)

#########

### CXCR5

CXCR5 <- ggplot(grouped\_median, aes(x = Condition, y = CXCR5) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"),

c("Day 3", "Day 5"), c("Day 3", "Discharge"),

c("Day 5", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

#run boxplot

CXCR5

ggsave(plot= CXCR5, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_MMI\_markers\_20231010\_statssig\_CXCR5.pdf", width = 5, height = 5, units = "in", dpi = 300)

####BOXPLOT FUNCTION TO MAKE MULTIPLE BOXPLOTS ON ALL MARKERS

boxplot\_fun = function(x, y) {

ggplot(grouped\_median, aes(x = .data[[y]], y = .data[[x]]) ) +

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =3, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#annotation\_logticks(scaled = TRUE, sides = "l", outside = FALSE)+

# scale\_y\_log10(expand = c(0, 0), limits=c(10^0, 10^6), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black"),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"),

c("Day 3", "Day 5"), c("Day 3", "Discharge"),

c("Day 5", "Discharge")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, label = "p.signif"

)

# theme(aspect.ratio = 5)

#stat\_summary(fun.y = median, geom="line", group= "Cohort", size = 2, color = "black")

}

all\_plots = map(expl,

~map(response, boxplot\_fun, y = .x) )

all\_plots$Condition[1:2]

pdf("Figures/SCRIPT 1/population boxplots/CD8T\_cell\_boxplots\_MMI\_markers\_202306012\_statssig\_ALL.pdf")

all\_plots

dev.off()

library(ggplot2)

library(ggpubr)

library(plyr)

library(dplyr)

library(ggplot2)

library(scales)

library(reshape2)

library(RColorBrewer)

library(destiny)

library(uwot)

library(slingshot)

library(ggrepel)

library(cowplot)

library(heatmaply)

library(tidyverse)

library(ggbeeswarm)

library(wesanderson)

library(purrr)

library(rstatix)

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

##IF NEEDED LOAD FILE

counts\_percofsample\_spread <- read\_csv("csv output/script 1/CD4\_T\_CELL\_frequency\_wide\_9\_clusters\_20231005.csv")

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

######### BOXPLOT PROPORTIONS ##############

#try a loop for proportions

colnames(counts\_percofsample\_spread)

#variables

counts\_percofsample\_spread <- counts\_percofsample\_spread[ -c(1) ]

head(counts\_percofsample\_spread)

response = names(counts\_percofsample\_spread)[6:14] #this is columns that will be on y axis

response = set\_names(response)

expl = names(counts\_percofsample\_spread)[5] #this is what column will be on x axis

expl = set\_names(expl)

#set the order of condition groups starting with healthy, admission, day3, day5 and discharge

counts\_percofsample\_spread$Condition <- factor(counts\_percofsample\_spread$Condition, levels=c('Healthy', 'Admission', 'Day 3', 'Day 5', 'Discharge'))

##### Make a list to use as a comparison i.e Healthy vs other conditions.

healthy\_vs <- list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"))

Admission\_vs <- list(c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"))

Day\_3\_vs <- list(c("Day 3", "Day 5"), c("Day 3", "Discharge"))

Day\_5\_vs <- list( c("Day 5", "Discharge"))

#####Check if the data is normally distributed

shapiro.test(counts\_percofsample\_spread$'1'[counts\_percofsample\_spread$Condition=="Healthy"]) #not normal

shapiro.test(counts\_percofsample\_spread$'1'[counts\_percofsample\_spread$Condition=="Admission"]) #not normal

shapiro.test(counts\_percofsample\_spread$'1'[counts\_percofsample\_spread$Condition=="Day 3"]) #not normal

shapiro.test(counts\_percofsample\_spread$'1'[counts\_percofsample\_spread$Condition=="Day 5"]) #not normal

shapiro.test(counts\_percofsample\_spread$'1'[counts\_percofsample\_spread$Condition=="Discharge"]) #not normal

shapiro.test(counts\_percofsample\_spread$'2'[counts\_percofsample\_spread$Condition=="Healthy"]) #not normal

shapiro.test(counts\_percofsample\_spread$'2'[counts\_percofsample\_spread$Condition=="Admission"]) #not normal

shapiro.test(counts\_percofsample\_spread$'2'[counts\_percofsample\_spread$Condition=="Day 3"]) #not normal

shapiro.test(counts\_percofsample\_spread$'2'[counts\_percofsample\_spread$Condition=="Day 5"]) #not normal

shapiro.test(counts\_percofsample\_spread$'2'[counts\_percofsample\_spread$Condition=="Discharge"]) #not normal

shapiro.test(counts\_percofsample\_spread$'3'[counts\_percofsample\_spread$Condition=="Healthy"]) #not normal

shapiro.test(counts\_percofsample\_spread$'3'[counts\_percofsample\_spread$Condition=="Admission"]) #not normal

shapiro.test(counts\_percofsample\_spread$'3'[counts\_percofsample\_spread$Condition=="Day 3"]) #normal

shapiro.test(counts\_percofsample\_spread$'3'[counts\_percofsample\_spread$Condition=="Day 5"]) #normal

shapiro.test(counts\_percofsample\_spread$'3'[counts\_percofsample\_spread$Condition=="Discharge"]) #normal

shapiro.test(counts\_percofsample\_spread$'4'[counts\_percofsample\_spread$Condition=="Healthy"]) #not normal

shapiro.test(counts\_percofsample\_spread$'4'[counts\_percofsample\_spread$Condition=="Admission"]) #not normal

shapiro.test(counts\_percofsample\_spread$'4'[counts\_percofsample\_spread$Condition=="Day 3"]) #not normal

shapiro.test(counts\_percofsample\_spread$'4'[counts\_percofsample\_spread$Condition=="Day 5"]) #not normal

shapiro.test(counts\_percofsample\_spread$'4'[counts\_percofsample\_spread$Condition=="Discharge"]) #not normal

shapiro.test(counts\_percofsample\_spread$'5'[counts\_percofsample\_spread$Condition=="Healthy"]) #normal

shapiro.test(counts\_percofsample\_spread$'5'[counts\_percofsample\_spread$Condition=="Admission"]) #normal

shapiro.test(counts\_percofsample\_spread$'5'[counts\_percofsample\_spread$Condition=="Day 3"]) #normal

shapiro.test(counts\_percofsample\_spread$'5'[counts\_percofsample\_spread$Condition=="Day 5"]) #normal

shapiro.test(counts\_percofsample\_spread$'5'[counts\_percofsample\_spread$Condition=="Discharge"]) #normal

shapiro.test(counts\_percofsample\_spread$'6'[counts\_percofsample\_spread$Condition=="Healthy"]) #normal

shapiro.test(counts\_percofsample\_spread$'6'[counts\_percofsample\_spread$Condition=="Admission"]) #normal

shapiro.test(counts\_percofsample\_spread$'6'[counts\_percofsample\_spread$Condition=="Day 3"]) #normal

shapiro.test(counts\_percofsample\_spread$'6'[counts\_percofsample\_spread$Condition=="Day 5"]) #normal

shapiro.test(counts\_percofsample\_spread$'6'[counts\_percofsample\_spread$Condition=="Discharge"]) #normal

shapiro.test(counts\_percofsample\_spread$'7'[counts\_percofsample\_spread$Condition=="Healthy"]) #not normal

shapiro.test(counts\_percofsample\_spread$'7'[counts\_percofsample\_spread$Condition=="Admission"]) #not normal

shapiro.test(counts\_percofsample\_spread$'7'[counts\_percofsample\_spread$Condition=="Day 3"]) #normal

shapiro.test(counts\_percofsample\_spread$'7'[counts\_percofsample\_spread$Condition=="Day 5"]) #normal

shapiro.test(counts\_percofsample\_spread$'7'[counts\_percofsample\_spread$Condition=="Discharge"]) #not normal

shapiro.test(counts\_percofsample\_spread$'9'[counts\_percofsample\_spread$Condition=="Healthy"]) #not normal

shapiro.test(counts\_percofsample\_spread$'9'[counts\_percofsample\_spread$Condition=="Admission"]) #not normal

shapiro.test(counts\_percofsample\_spread$'9'[counts\_percofsample\_spread$Condition=="Day 3"]) #not normal

shapiro.test(counts\_percofsample\_spread$'9'[counts\_percofsample\_spread$Condition=="Day 5"]) #not normal

shapiro.test(counts\_percofsample\_spread$'9'[counts\_percofsample\_spread$Condition=="Discharge"]) #not normal

shapiro.test(counts\_percofsample\_spread$'10'[counts\_percofsample\_spread$Condition=="Healthy"]) #not normal

shapiro.test(counts\_percofsample\_spread$'10'[counts\_percofsample\_spread$Condition=="Admission"]) #not normal

shapiro.test(counts\_percofsample\_spread$'10'[counts\_percofsample\_spread$Condition=="Day 3"]) #not normal

shapiro.test(counts\_percofsample\_spread$'10'[counts\_percofsample\_spread$Condition=="Day 5"]) #not normal

shapiro.test(counts\_percofsample\_spread$'10'[counts\_percofsample\_spread$Condition=="Discharge"]) #not normal

#rename cell population columns

counts\_percofsample\_label <- counts\_percofsample\_spread %>%

rename(ActEff = '1',

EM1 = '2',

EM2 = '3',

Treg\_memory1 = '4',

CM = '5',

Naive = '6',

ExNaive = '7',

Tregmem = '9',

ActNaive = '10')

head(counts\_percofsample\_label)

head(counts\_percofsample\_spread)

counts\_percofsample\_spread %>% shapiro\_test(healthy\_vs)

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

#################################################################3#

############ NEED TO THINK HOW TO MEASURE BOXPLOTS, NOT WORKING ANYMORE.

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

##Cluster 2 (Effector memory 1)

EM1 <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = EM1))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Effector memory 1")

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"),

c("Day 3", "Day 5"), c("Day 3", "Discharge"),

c("Day 5", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

EM1

ggsave(plot= EM1, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_EM1\_20231004.png", width = 5, height = 5, units = "in", dpi = 300)

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

## Cluster 1 (Acitvated effector)

ActEff <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = ActEff))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Activated Effector")

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"),

c("Day 3", "Day 5"), c("Day 3", "Discharge"),

c("Day 5", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

ActEff

ggsave(plot= EM, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_EM\_20231004.png", width = 5, height = 5, units = "in", dpi = 300)

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

## Cluster 3 (Effector memory 2)

EM2 <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = EM2))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Effector memory 2")

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"),

c("Day 3", "Day 5"), c("Day 3", "Discharge"),

c("Day 5", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

EM2

ggsave(plot= EM2, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_EM2\_20231004.png", width = 5, height = 5, units = "in", dpi = 300)

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

## cluster 7 (Exhausted naive)

ExNaive <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = ExNaive))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Exhausted Naive")+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

ExNaive

ggsave(plot= ExNaive, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_ExNaive\_20231009.png", width = 7, height = 7, units = "in", dpi = 300)

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

## Cluster 9 (Treg memory)

Tregmem <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = Tregmem))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Treg memory")+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Admission", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

Tregmem

ggsave(plot= Tregmem, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_Tregmem\_20231004.png", width = 5, height = 5, units = "in", dpi = 300)

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

## Cluster 5 (CM)

CM <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = CM))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Central memory")

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"),

c("Day 3", "Day 5"), c("Day 3", "Discharge"),

c("Day 5", "Discharge")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

CM

ggsave(plot= CM, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_CM\_20231004.png", width = 5, height = 5, units = "in", dpi = 300)

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

## Cluster 6 (Naive)

Naive <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = Naive))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Naive")+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

Naive

ggsave(plot= Naive, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_Naive\_20231009.png", width = 5, height = 5, units = "in", dpi = 300)

######

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

## cluster 4 (Treg memory 1)

Tregmem1 <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = Treg\_memory1))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Treg memory1")

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Day 3"), c("Healthy", "Day 5"), c("Healthy", "Discharge"),

c("Admission", "Day 3"), c("Admission", "Day 5"), c("Admission", "Discharge"),

c("Day 3", "Day 5"), c("Day 3", "Discharge"),

c("Day 5", "Discharge")),

method = "t.test", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

Tregmem1

ggsave(plot= Tregmem1, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_Tregmem1\_20231009.png", width = 5, height = 5, units = "in", dpi = 300)

###

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

## Clsuter 10 (ActNaive)

ActNaive <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = ActNaive))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("Activated Naive")+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Healthy", "Admission"), c("Healthy", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

ActNaive

ggsave(plot= ActNaive, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_Activatednaive(1)\_20231009.png", width = 5, height = 5, units = "in", dpi = 300)

###

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

##Double poostive transitional

Th2 <- ggplot(counts\_percofsample\_label, aes(x = Condition, y = Th2))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

scale\_y\_continuous("T helper 2")+

stat\_compare\_means(mapping = (aes(label=..p.sig..)),

comparisons = list(c("Admission", "Discharge")),

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

Th2

ggsave(plot= Th2, "Figures/SCRIPT 1/Population boxplots/CD4T\_cell\_boxplots\_prop\_TH2\_20231005.png", width = 5, height = 5, units = "in", dpi = 300)

#####

boxplot\_fun = function(x, y) {

ggplot(counts\_percofsample\_spread, aes(x = .data[[y]], y = .data[[x]]))+

geom\_boxplot(outlier.shape = NA)+

geom\_quasirandom(show.legend = NA, size =4, pch=21, aes(fill = Condition))+

scale\_fill\_manual(values = c("Healthy" = "#55a8ca",

"Admission" = "#ede151",

"Day 3" ="#f28e49",

"Day 5" = "#a65d86",

"Discharge" = "#3c398d"))+

#scale\_y\_log10(expand = c(0, 0), limits=c(10^1, 10^8), breaks = scales::trans\_breaks("log10", function(x) 10^x),

# labels = scales::trans\_format("log10", scales::math\_format(10^.x)))+

theme\_classic()+

theme(legend.position = "none")+

theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank())+

xlab("")+

theme(axis.title.x = element\_text(size=12, colour = "black"))+

theme(axis.title.y = element\_text(size=12, colour = "black"))+

theme(axis.text.x = element\_text(size=12, face= "bold", colour = "black", angle = 45, vjust =1, hjust=1),

axis.text.y = element\_text(size=12, face = "bold", colour = "black"))+

theme(legend.text=element\_text(size=12))+

stat\_compare\_means(mapping = (aes(label=..p.sig..)), comparisons = healthy\_vs,

method = "wilcox", paired = FALSE,

hide.ns = FALSE, #this will show if any p.values are not sig

label = "p.signif" #this is show the astrix, ##it out to see the p.value

)

}

colnames(counts\_percofsample\_spread)

boxplot\_fun

all\_plots = map(expl,

~map(response, boxplot\_fun, y = .x) )

all\_plots$Condition[1:2]

pdf("Figures/SCRIPT 1/population boxplots/CD4T\_cell\_boxplots\_prop\_9clusters\_20230601\_wilcox\_sig\_Healthy\_vs.pdf")

all\_plots

dev.off()