Figure 5.1 Data analysis script

Contents

make volcano plot


```
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=FALSE)
```

1 load the packages

2 Set up the data

2.1 Import df, Annotation file & Kallisto sc run metrics

```
## [1] 34963   480
```

```
## [1] 480   9
```

2.2 subset the data, only consider single cells

```
sc_Annotation_12345 <- subset(sorted_Annotation_12345, Cell.Number == 1)
sc_Annotation_12345 <- droplevels(sc_Annotation_12345)
dim(sc_Annotation_12345)
```

```
## [1] 437   9
```

```
mc_Annotation_12345 <- subset(sorted_Annotation_12345, Cell.Number != 1)
mc_Annotation_12345 <- droplevels(mc_Annotation_12345)
```

```
sc_Tag_12345 <- TagPL12345_genes[,colnames(TagPL12345_genes) %in% rownames(
        sc_Annotation_12345)]
dim(sc_Tag_12345)
```

```
## [1] 34963   437
```

```
mc_Tag_12345 <- TagPL12345_genes[,colnames(TagPL12345_genes) %in% rownames(
        mc_Annotation_12345)]
dim(mc_Tag_12345)
```

```
## [1] 34963    43
```

```
Kallisto_run_metrics_Tag12345_sc <- Kallisto_run_metrics_Tag12345[
        rownames(Kallisto_run_metrics_Tag12345)%in%
                rownames(sc_Annotation_12345),]
dim(Kallisto_run_metrics_Tag12345_sc)
```

```
## [1] 437   3
```

2.3 Subset on Vaccinated samples for scatter plots to explore cellular heterogenity

```
Vac_ann <- subset(sc_Annotation_12345, Type == 'Vaccinated')
V_df <- sc_Tag_12345[,colnames(sc_Tag_12345)%in% rownames(Vac_ann)]
V_df <- as.data.frame(V_df)

Vac_mc_ann <- subset(mc_Annotation_12345, Type == 'Vaccinated')
V_mc_df <- mc_Tag_12345[,colnames(mc_Tag_12345)%in% rownames(Vac_mc_ann)]
V_mc_df <- as.data.frame(V_mc_df)
```

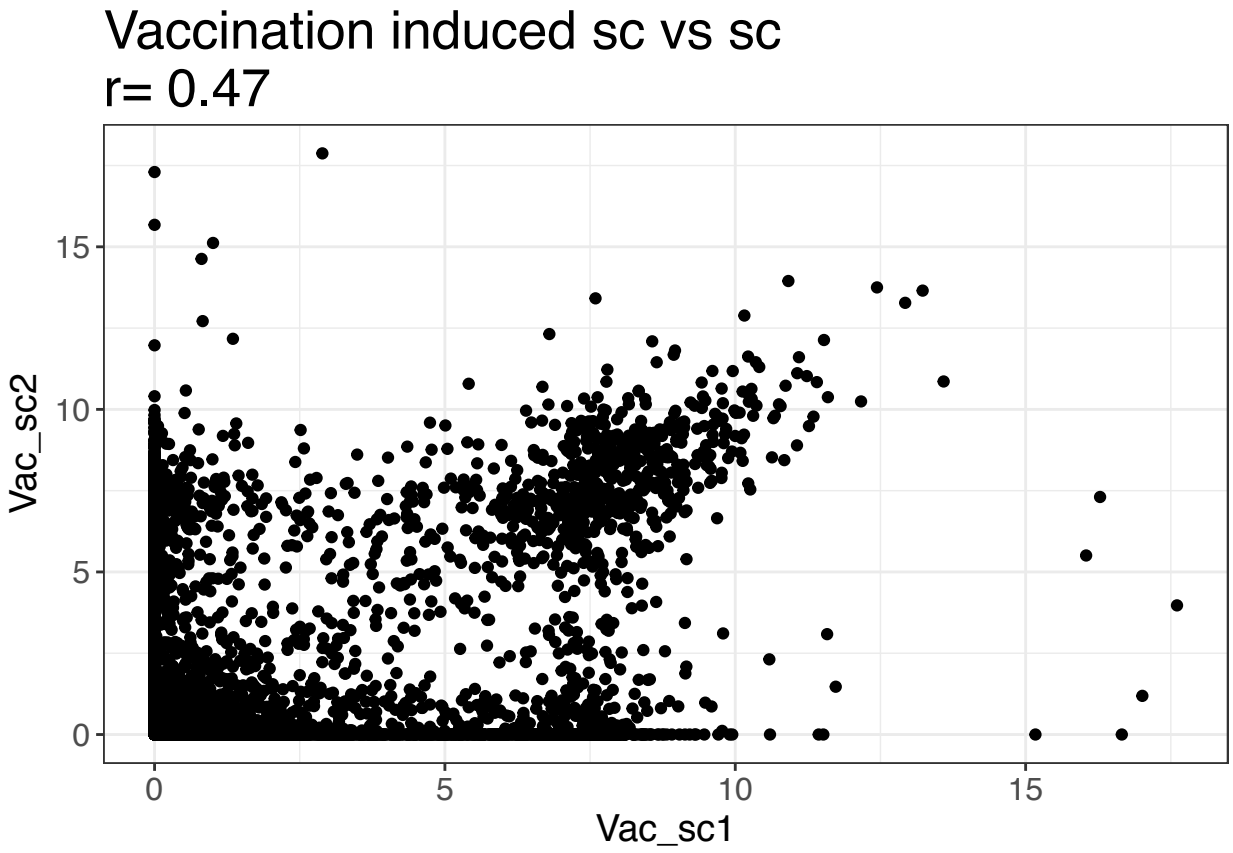
2.4 Functions I use to make scatter plots

```
scatter_func <- function(data, xvar, yvar){
  g <-ggplot(data, aes_string(x = xvar, y = yvar)) +
    geom_point() +
          theme_bw() +
          theme(axis.text=element_text(size=12),
                axis.title=element_text(size=14),
                plot.title =element_text(size=20))
g
}
pearsoncorr <- function(xvar, yvar){
        Number <- cor(xvar, yvar, use= 'na.or.complete')
        Value <- signif(Number, digits=2)
        Value
}
Title <- function(main, pearson){
  mainTitle <- paste(main, toString(pearson))
  mainTitle
}
```

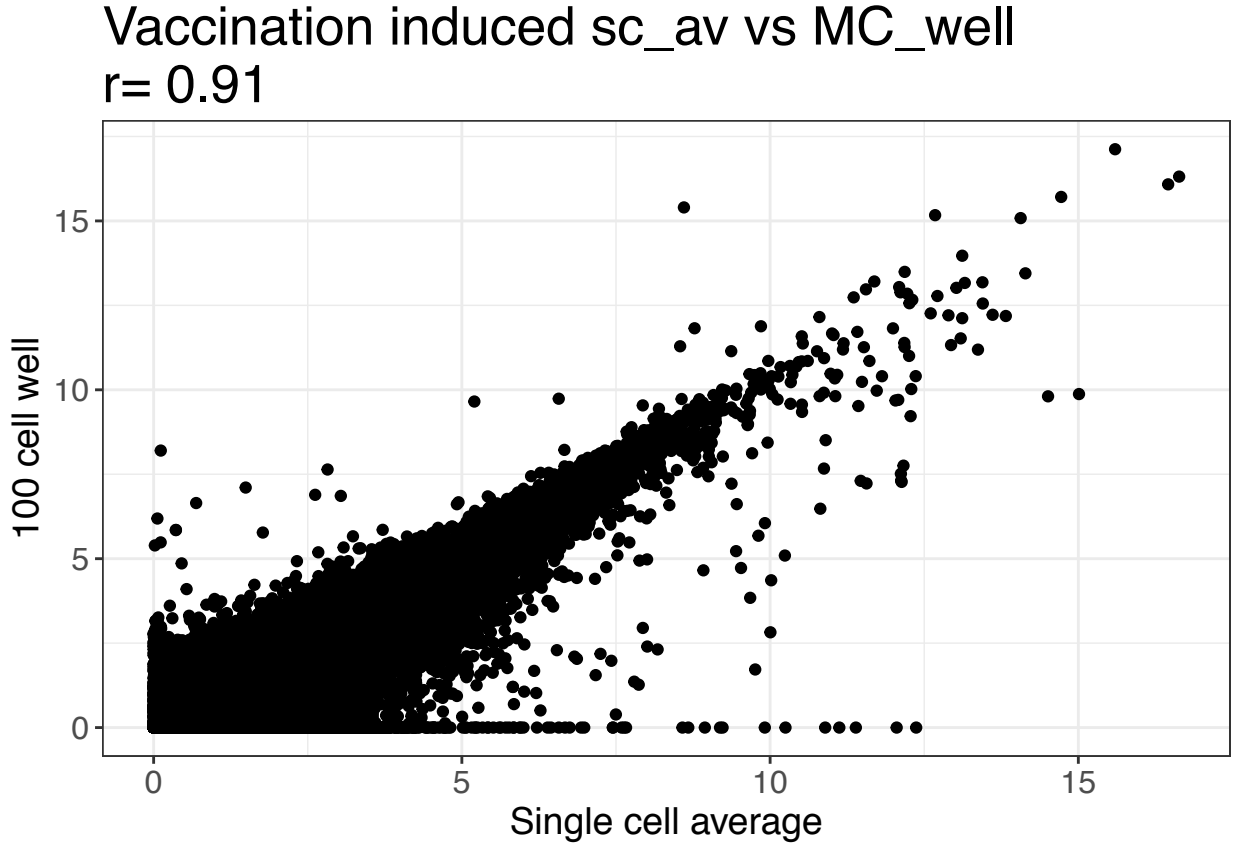
2.5 Generating the plots

```
#Vac sc vs Vac sc
```

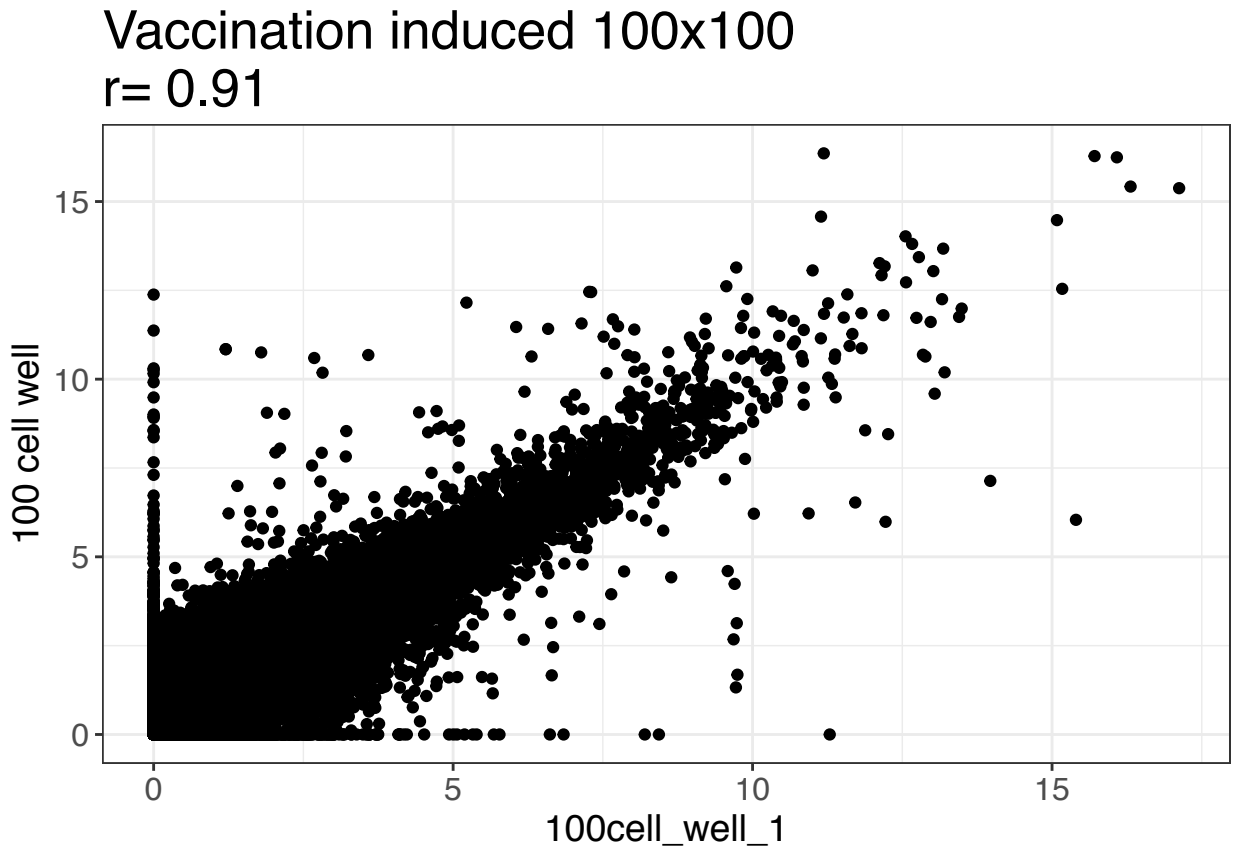
```
new_df <- as.data.frame(cbind(log2(V_df$`PW2-A1`+1), log2(V_df$`PW6-H6`+1)))
colnames(new_df) <- c("Vac_sc1", "Vac_sc2")
R <- pearsoncorr(new_df$Vac_sc1, new_df$Vac_sc2)
maintitle <- Title("Vaccination induced sc vs sc \nr=", R)
plot <- scatter_func(new_df, "Vac_sc1", "Vac_sc2")
g <- plot + labs(x = "Vac_sc1", y = "Vac_sc2") + ggtitle(maintitle)
g
```




```
#Vac sc average vs Vac 100 cell well
Av_V_DF <- rowMeans(V_df, na.rm=TRUE)
log_Av_V_DF <- (log2(Av_V_DF +1))
new_df <- as.data.frame(cbind(log_Av_V_DF, log2(V_mc_df$`PW2-A9`+1)))
colnames(new_df) <- c("single_cell_av", "MC-cell_well")
R <- pearsoncorr(new_df$single_cell_av, new_df$`MC-cell_well`)
maintitle <- Title("Vaccination induced sc_av vs MC_well \nr=", R)
plot <- scatter_func(new_df, "single_cell_av", new_df$`MC-cell_well`)
g <- plot + labs(x = "Single cell average", y = "100 cell well") +
        ggtitle(maintitle)
g
```




```
#Vac 100 cell well vs Vac 100 cell well
new_df <- as.data.frame(cbind(log2(V_mc_df$`PW2-A9`+1),
                              log2(V_mc_df$`PW4-G9`+1)))
colnames(new_df) <- c("MCcell_well_1", "MCcell_well")
R <- pearsoncorr(new_df$MCcell_well_1, new_df$MCcell_well)
maintitle <- Title("Vaccination induced 100x100 \nr=", R)
plot <- scatter_func(new_df, new_df$MCcell_well_1, new_df$MCcell_well)
g <- plot + labs(x = "100cell_well_1", y = "100 cell well") +
        ggtitle(maintitle)
g
```



##log2 transform

```
Log2_Tag_12345_sc <- log2(sc_Tag_12345 + 1)
dim(Log2_Tag_12345_sc)
```

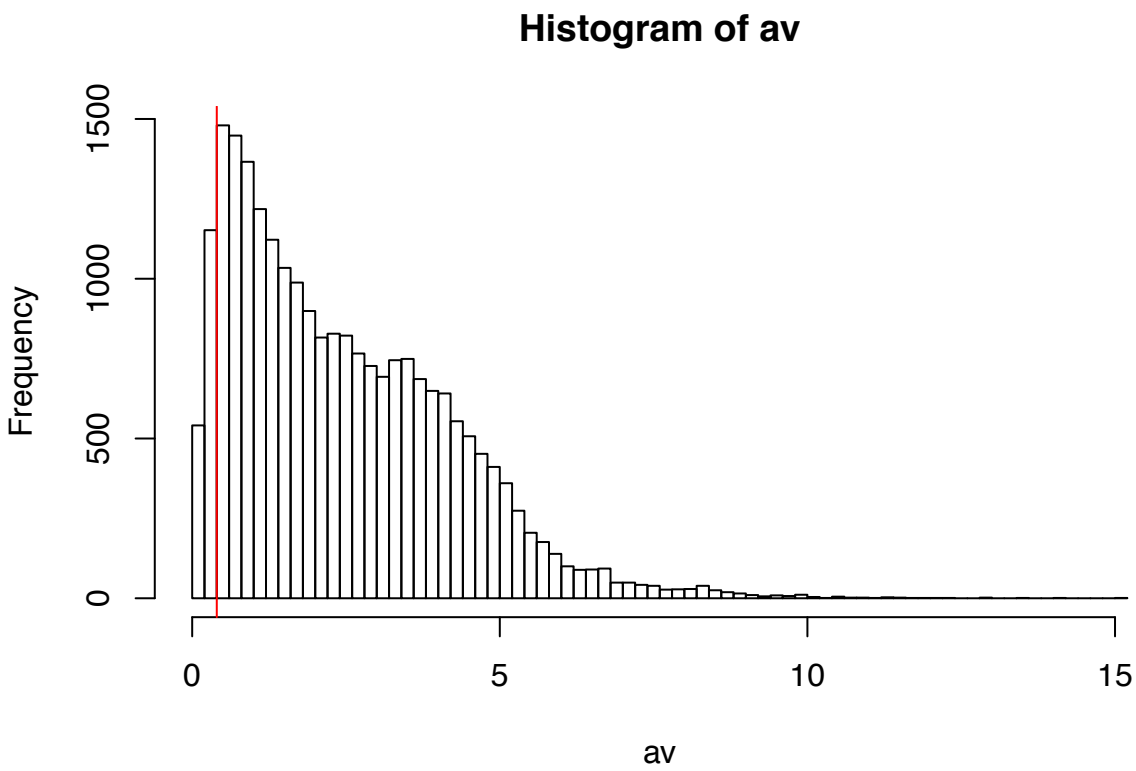
```
## [1] 34963   437
```

```
Log2_Tag_12345_mc <- log2(mc_Tag_12345 + 1)
dim(Log2_Tag_12345_mc)
```

```
## [1] 34963    43
```

2.6 Here find the sc threshold and set values below it = 0

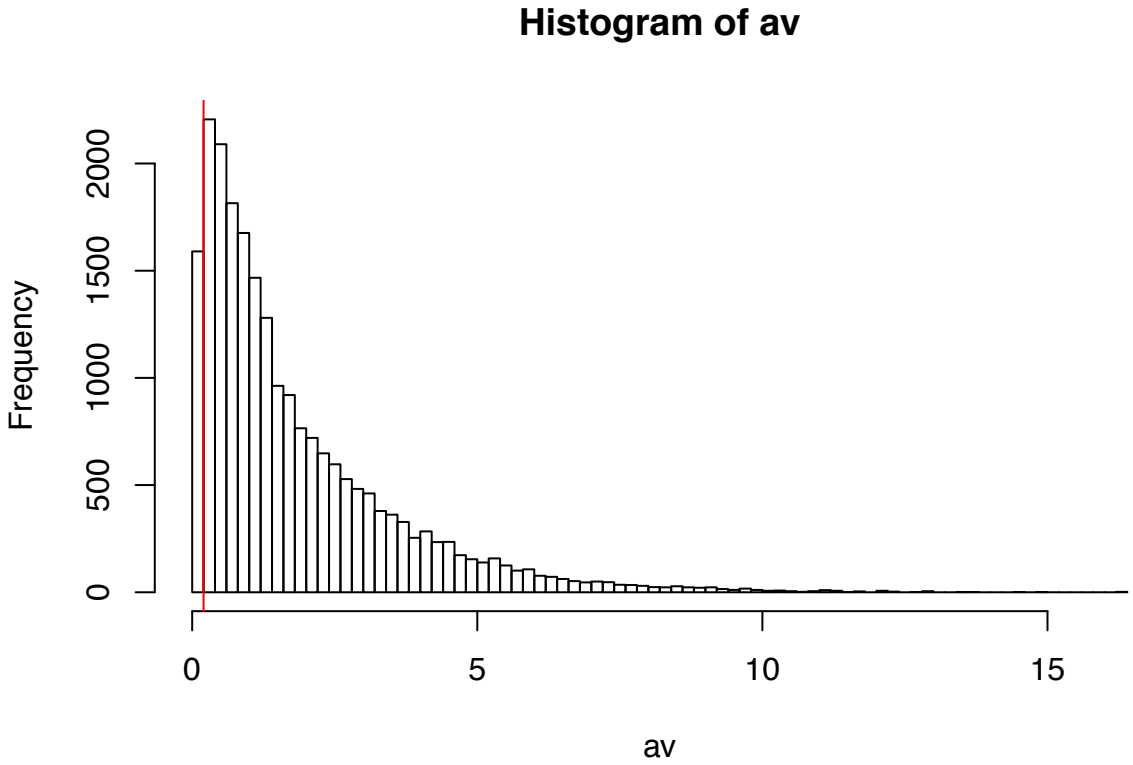
```
g <- Log2_Tag_12345_sc
g[g ==0] <-NA
av <- rowMeans(g, na.rm = TRUE)
hist(av, breaks = 100)
abline(v=.4, col= 'red')
```




```
Threshold_Log2_Tag_12345_sc <- Log2_Tag_12345_sc
Threshold_Log2_Tag_12345_sc[Threshold_Log2_Tag_12345_sc < .4] <- 0
```

2.7 Same for mc data

```
g <- Log2_Tag_12345_mc
g[g ==0] <-NA
av <- rowMeans(g, na.rm = TRUE)
hist(av, breaks = 100)
abline(v=.2, col= 'red')
```




```
Threshold_Log2_Tag_12345_mc <- Log2_Tag_12345_mc
Threshold_Log2_Tag_12345_mc[Threshold_Log2_Tag_12345_mc < .2] <- 0
```

2.8 Re-merge the data sets and Remove genes with variance = 0

```
Full_DF_again <- cbind(Threshold_Log2_Tag_12345_sc,
                       Threshold_Log2_Tag_12345_mc)
Full_DF_variance <- (apply(Full_DF_again, 1, var, na.rm = TRUE) == 0)
var_Full_DF <- Full_DF_again[!Full_DF_variance,]
dim(var_Full_DF)
```

```
## [1] 23974   480
```

2.9 Remove duplicated elements in Gene list.

2.9.1 These are situations where there are multiple names for a given gene.

```
dups <- duplicated(var_Full_DF)
dup_var_Full_DF <- var_Full_DF[!dups,]
dim(dup_var_Full_DF)
```

```
## [1] 23904   480
```

3 Subset on Vaccine induced samples and apply cell filters

3.1 Separate Vaccine induced samples from the log transformed data here.

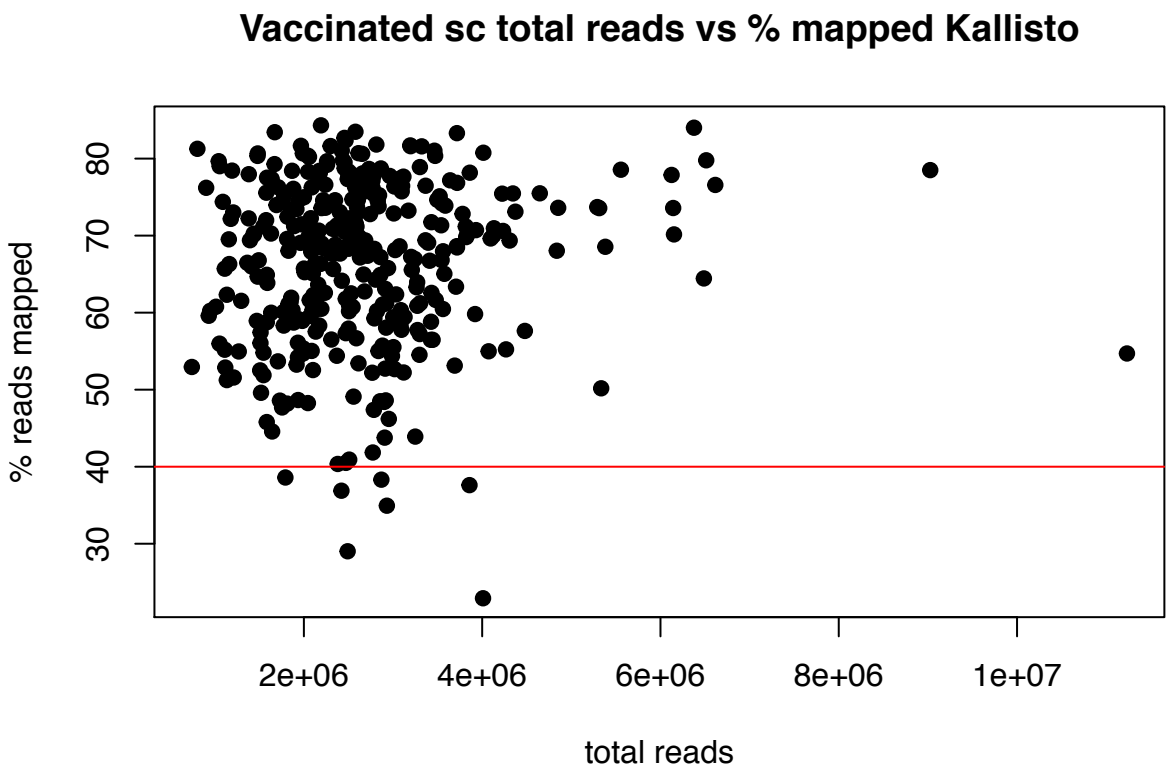
```
Vaccinated_ann <- subset(sc_Annotation_12345, Type == 'Vaccinated')
Vaccinated_DF <- dup_var_Full_DF[,colnames(dup_var_Full_DF)%in%rownames(
        Vaccinated_ann)]
dim(Vaccinated_DF)
```

```
## [1] 23904   350
```

```
Vaccinated_Kallisto_run_Metrics <- Kallisto_run_metrics_Tag12345[rownames(
        Kallisto_run_metrics_Tag12345)
        %in%rownames(Vaccinated_ann),]
```

3.2 Make % mapping figure and apply filter

```
plot(Vaccinated_Kallisto_run_Metrics$processed.reads,
     Vaccinated_Kallisto_run_Metrics$X..mapped.reads,
     main="Vaccinated sc total reads vs % mapped Kallisto",
     ylab = "% reads mapped", xlab="total reads", pch = 19)
abline(h=40, col='red')
```




```
PercentMapped_filter_Vaccinated <- subset(Vaccinated_Kallisto_run_Metrics,
                                          X..mapped.reads > 40)

Vaccinated_DF_Filtered_Kallisto <- Vaccinated_DF[,(colnames(
        Vaccinated_DF) %in% rownames(PercentMapped_filter_Vaccinated))]
dim(Vaccinated_DF_Filtered_Kallisto)
```

```
## [1] 23904   343
```

```
Filtered_Vaccinated_ann <- Vaccinated_ann[rownames(
        Vaccinated_ann)%in%colnames(Vaccinated_DF_Filtered_Kallisto),]
```

3.3 Import repertoire and remove all cells with unknown specificity

```
notExclude <- subset(Vaccine_repertoire, HC.Seq.Source != 'exclude')
Vac_repertoire_notExclude <- Vaccine_repertoire[rownames(
        Vaccine_repertoire)%in% rownames(notExclude),]
dim(Vac_repertoire_notExclude) #lost 13 cells
```

```
## [1] 330  41
```

```
Vac_repertoire_notExclude <- droplevels(Vac_repertoire_notExclude)
```

3.4 Remove any IgG vaccine negative cells that snuck through

```
Vac_IGneg_ANN <-subset(Filtered_Vaccinated_ann,
                       Isotype == 'IgG' & Specificity == 'negative')
dim(Vac_IGneg_ANN)
```

```
## [1] 6 9
```

3.5 Filter DF on remaining cells

```
Vaccinated_Filtered_sc_Kallisto_Rep_DF <- Vaccinated_DF_Filtered_Kallisto[
        ,colnames(
        Vaccinated_DF_Filtered_Kallisto) %in%
                rownames(Vac_repertoire_notExclude)]

Vaccinated_Filtered_sc_Kallisto_Rep_DF <-
        Vaccinated_Filtered_sc_Kallisto_Rep_DF[
                ,!colnames(Vaccinated_Filtered_sc_Kallisto_Rep_DF) %in%
                        rownames(Vac_IGneg_ANN)]
```

3.6 Make new annotation files to exclude the lost cells

```
filtered_sc_Annotation_Vaccinated <- Filtered_Vaccinated_ann[
        rownames(Filtered_Vaccinated_ann) %in%
                colnames(Vaccinated_Filtered_sc_Kallisto_Rep_DF),]
filtered_sc_Annotation_Vaccinated <- droplevels(
        filtered_sc_Annotation_Vaccinated)
dim(filtered_sc_Annotation_Vaccinated)
```

```
## [1] 324   9
```

```
Filtered_repertoire_vaccinated <- Vac_repertoire_notExclude[
        rownames(Vac_repertoire_notExclude)%in%
                colnames(Vaccinated_Filtered_sc_Kallisto_Rep_DF),]
dim(Filtered_repertoire_vaccinated)
```

```
## [1] 324  41
```

3.7 Remove cells with specificity = 'exclude' or 'failure'

```
Skip <- subset(filtered_sc_Annotation_Vaccinated, Specificity == 'exclude'
               | Specificity == 'failure')
dim(Skip) #29 cells exclded
```

```
## [1] 29  9
```

```
Vac_annotation_known <- filtered_sc_Annotation_Vaccinated[!rownames(
        filtered_sc_Annotation_Vaccinated)%in%rownames(Skip),]
Vac_annotation_known <- droplevels(Vac_annotation_known)
dim(Vac_annotation_known)
```

```
## [1] 295   9
```

```
Vac_repertoire_known <- Filtered_repertoire_vaccinated[
        rownames(Filtered_repertoire_vaccinated)%in%
                rownames(Vac_annotation_known),]
Vac_repertoire_known <- droplevels(Vac_repertoire_known)
dim(Vac_repertoire_known)
```

```
## [1] 295  41
```

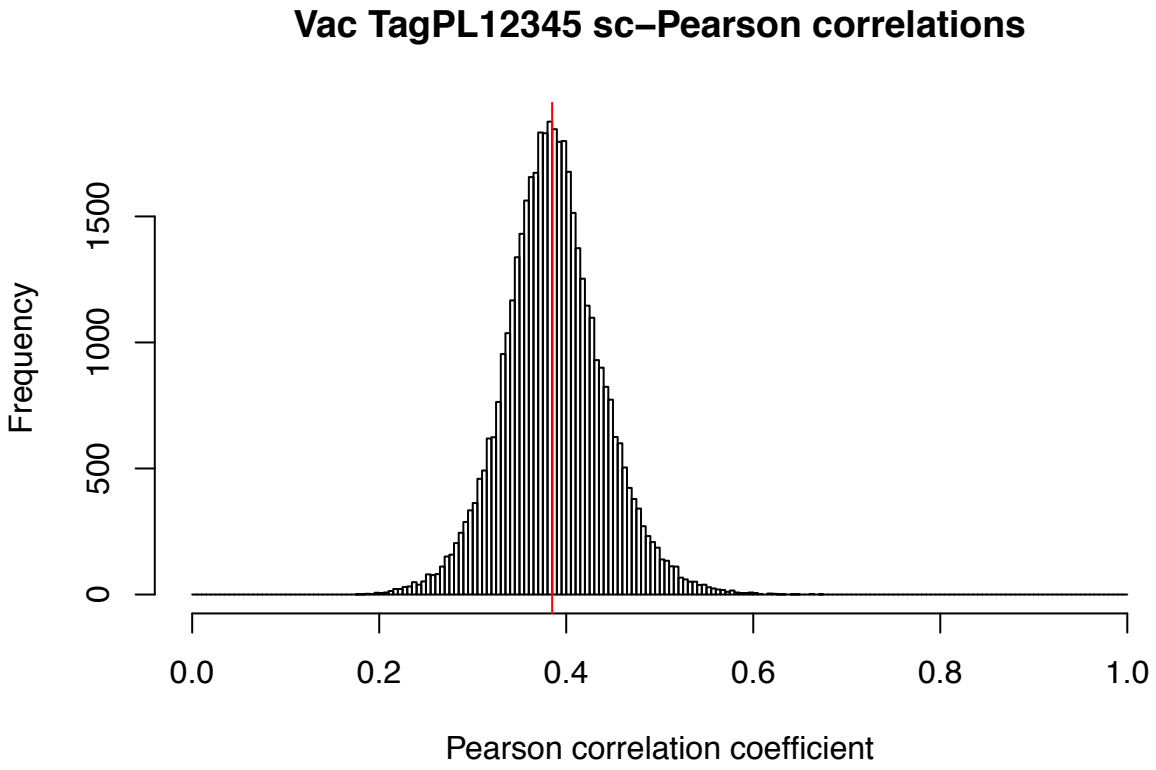
```
Vac_DF_known <- Vaccinated_Filtered_sc_Kallisto_Rep_DF[
        ,colnames(Vaccinated_Filtered_sc_Kallisto_Rep_DF)%in%
                rownames(Vac_annotation_known)]
dim(Vac_DF_known)
```

```
## [1] 23904   295
```

3.8 Calculate pearson itteration to reveal distribution of heterogenity between single cells

```
pearson_itteration <- function(df)
        {
        p = list()
        for (i in 1:(ncol(df))){
                for (k in (i + 1):(ncol(df))){
                        if (i < (ncol(df))){
                                pcor <- cor(df[i], df[k], use= "na.or.complete")
                                #print(pcor)
                                p <- c(p, pcor)
                        }
                }
        }
        p
}

Vac_TagPL12345_Pearson = pearson_itteration(Vac_DF_known)
testn = as.numeric(Vac_TagPL12345_Pearson)
middle = median(testn)
hist(as.numeric(Vac_TagPL12345_Pearson), breaks=seq(0,1,by=.005),
```

```
main="Vac TagPL12345 sc-Pearson correlations",
xlab = "Pearson correlation coefficient")
   abline(v=middle, col='red')
```



4 Apply gene filter, normalize the data and remove batch effects

4.1 Now apply gene expression filter, where expression is required in 5% of the cells

```
morethanNnonzero <- function(x,N){
   y <- x[rowSums(x!=0)>=N,]
   y
}
dim(Vac_DF_known)
```

```
## [1] 23904   295
```

```
Filtered_TagPL12345_Vaccine_DF_sc <- morethanNnonzero(Vac_DF_known,(.05*295))
dim(Filtered_TagPL12345_Vaccine_DF_sc)
```

```
## [1] 11895   295
```

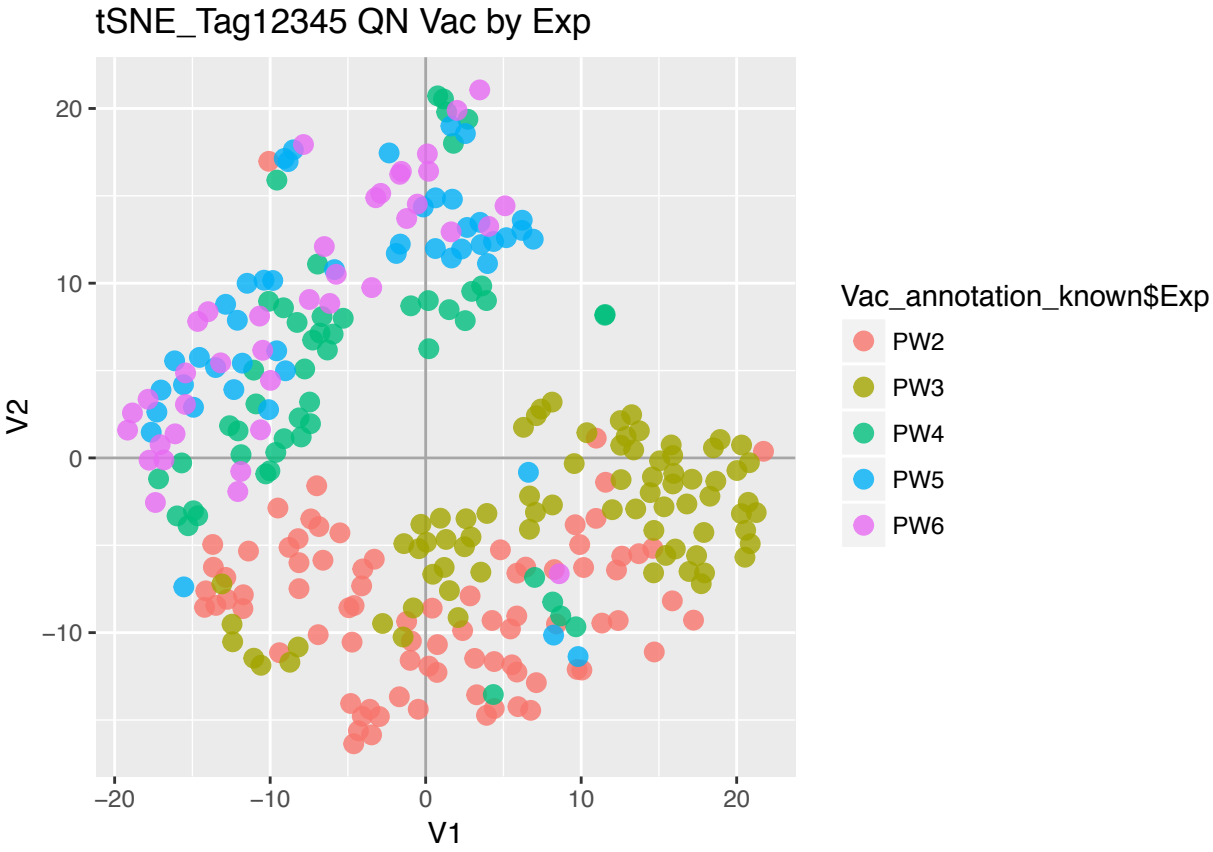
4.2 Quantile normalize sc DF by experimental plate.

```
library(limma)
library(sva)
#install.packages("preprocessCore")
library(preprocessCore)
exp_PW2 <- subset(Vac_annotation_known, Exp == "PW2")
exp_PW3 <- subset(Vac_annotation_known, Exp == "PW3")
exp_PW4 <- subset(Vac_annotation_known, Exp == "PW4")
exp_PW5 <- subset(Vac_annotation_known, Exp == "PW5")
exp_PW6 <- subset(Vac_annotation_known, Exp == "PW6")
PW2_post5EXP <- Filtered_TagPL12345_Vaccine_DF_sc[,colnames(
        Filtered_TagPL12345_Vaccine_DF_sc)%in%rownames(exp_PW2)]
PW3_post5EXP <- Filtered_TagPL12345_Vaccine_DF_sc[,colnames(
        Filtered_TagPL12345_Vaccine_DF_sc)%in%rownames(exp_PW3)]
PW4_post5EXP <- Filtered_TagPL12345_Vaccine_DF_sc[,colnames(
        Filtered_TagPL12345_Vaccine_DF_sc)%in%rownames(exp_PW4)]
PW5_post5EXP <- Filtered_TagPL12345_Vaccine_DF_sc[,colnames(
        Filtered_TagPL12345_Vaccine_DF_sc)%in%rownames(exp_PW5)]
PW6_post5EXP <- Filtered_TagPL12345_Vaccine_DF_sc[,colnames(
        Filtered_TagPL12345_Vaccine_DF_sc)%in%rownames(exp_PW6)]
qn_PW2_5 <- as.data.frame(normalize.quantiles(as.matrix(PW2_post5EXP),
                                              copy=TRUE))
rownames(qn_PW2_5) <- rownames(PW2_post5EXP)
colnames(qn_PW2_5) <- colnames(PW2_post5EXP)
qn_PW3_5 <- as.data.frame(normalize.quantiles(as.matrix(PW3_post5EXP),
                                              copy=TRUE))
rownames(qn_PW3_5) <- rownames(PW3_post5EXP)
colnames(qn_PW3_5) <- colnames(PW3_post5EXP)
qn_PW4_5 <- as.data.frame(normalize.quantiles(as.matrix(PW4_post5EXP),
                                              copy=TRUE))
rownames(qn_PW4_5) <- rownames(PW4_post5EXP)
colnames(qn_PW4_5) <- colnames(PW4_post5EXP)
qn_PW5_5 <- as.data.frame(normalize.quantiles(as.matrix(PW5_post5EXP),
                                              copy=TRUE))
rownames(qn_PW5_5) <- rownames(PW5_post5EXP)
colnames(qn_PW5_5) <- colnames(PW5_post5EXP)
qn_PW6_5 <- as.data.frame(normalize.quantiles(as.matrix(PW6_post5EXP),
                                              copy=TRUE))
rownames(qn_PW6_5) <- rownames(PW6_post5EXP)
colnames(qn_PW6_5) <- colnames(PW6_post5EXP)
QN_Tag12345_Vac_sc <- cbind(qn_PW2_5,qn_PW3_5,qn_PW4_5, qn_PW5_5, qn_PW6_5)
dim(QN_Tag12345_Vac_sc)
```

```
## [1] 11895   295
```

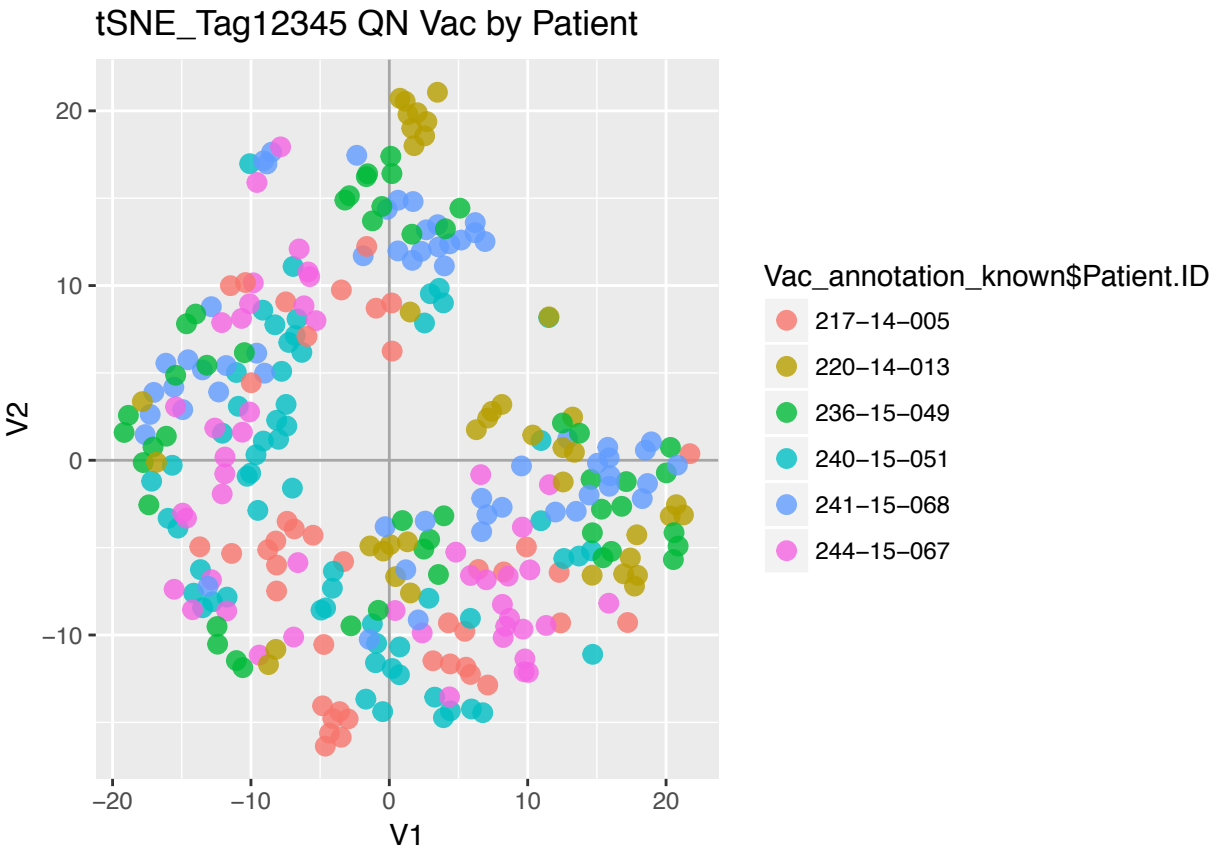
4.3 tSNE before limma to show batch effect

```
library(Rtsne)
set.seed(27)
trans_QN_Tag12345_Vac <- as.data.frame(t(QN_Tag12345_Vac_sc))
tSNE_Tag12345_QN_Vac <- Rtsne(data.frame(trans_QN_Tag12345_Vac),
                              check_duplicates = FALSE, theta = 0.001,
                              perplexity = 30,
                              initial_dims = 10)
ID <- ggplot(data = as.data.frame(tSNE_Tag12345_QN_Vac$Y),
             aes(x=V1, y=V2,label = colnames(QN_Tag12345_Vac_sc)))   +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vac_annotation_known$Exp),
                   alpha = 0.8, size = 3) +
        ggtitle("tSNE_Tag12345 QN Vac by Exp")
ID
```




```
ID <- ggplot(data = as.data.frame(tSNE_Tag12345_QN_Vac$Y),
             aes(x=V1, y=V2,label = colnames(QN_Tag12345_Vac_sc)))   +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vac_annotation_known$Patient.ID),
```

```
                alpha = 0.8, size = 3) +
        ggtitle("tSNE_Tag12345 QN Vac by Patient")
ID
```



4.4 Use Limma to remove batch effect from experimental plate and Patient ID, while protecting Isotype and Specificity

```
#source("https://bioconductor.org/biocLite.R")
#biocLite("limma")
library(limma)
Iso_Spec <- model.matrix(~0+Vac_annotation_known$Isotype+
                                 Vac_annotation_known$Specificity)
Limma_adj_4 <- removeBatchEffect(QN_Tag12345_Vac_sc,
                                 batch=Vac_annotation_known$Exp,
                                 batch2 = Vac_annotation_known$Patient.ID,
                                 design = Iso_Spec)
Limma_adj_4 <- as.data.frame(Limma_adj_4)
```

5 Begin to explore the data

5.1 Make a heatmap of the IGH-Constant domains to confirm that the identified BCR isotype matches the RNA expression levels.

5.1.1 1st - Make color spectrum for heatmap

```
col_spectrum <- colorRampPalette(brewer.pal(11,"RdBu"))(200)
```

5.1.2 2nd - Set up annotation for heatmap

```
IGH_list <- c('IGHA1', 'IGHA2', 'IGHG1','IGHG2', 'IGHG3', 'IGHG4')
Vaccine_IGHC_genes <- Limma_adj_4[rownames(Limma_adj_4)%in% IGH_list,]
dim(Vaccine_IGHC_genes)
```

```
## [1]   6 295
```

```
#Make Annotation
Flow_Iso_12345_V <- Vac_repertoire_known$HC.Isotype
summary(Flow_Iso_12345_V)
```

```
## IgA IgG
## 177 118
```

```
Flow_Iso_12345_V_ann<- gsub("IgA", "blue", Flow_Iso_12345_V)
Flow_Iso_12345_V_ann<- gsub("IgG", "green", Flow_Iso_12345_V_ann)

Sanger_Iso_12345_V <- Vac_repertoire_known$X2ndPCR.Isotype
summary(Sanger_Iso_12345_V)
```

```
## IgA Family       IgA1 IgG Family       IgG1       IgG2       IgG3
##         49        111          4        104          1          3
##    unclear
##         23
```

```
Sanger_Iso_12345_V_ann<- gsub("IgA1", "skyblue1", Sanger_Iso_12345_V)
Sanger_Iso_12345_V_ann<- gsub("IgA Family", "blue", Sanger_Iso_12345_V_ann)
Sanger_Iso_12345_V_ann<- gsub("IgA2", "royalblue1", Sanger_Iso_12345_V_ann)
Sanger_Iso_12345_V_ann<- gsub("IgG1", "darkgreen", Sanger_Iso_12345_V_ann)
Sanger_Iso_12345_V_ann<- gsub("IgG Family", "green", Sanger_Iso_12345_V_ann)
Sanger_Iso_12345_V_ann<- gsub("IgG2", "darkolivegreen1",
                              Sanger_Iso_12345_V_ann)
Sanger_Iso_12345_V_ann<- gsub("IgG3", "yellowgreen", Sanger_Iso_12345_V_ann)
Sanger_Iso_12345_V_ann<- gsub("unclear", "grey", Sanger_Iso_12345_V_ann)
```

```
Algorithm_Iso_12345_V <- Vac_repertoire_known$BASIC.Isotype
summary(Algorithm_Iso_12345_V)
```

```
##    IgA Family          IgA1          IgA2    IgG Family          IgG1
##             3           143            29             1           111
##          IgG2          IgG3 not available       unclear
##             1             4             2             1
```

```
Algorithm_Iso_12345_V_ann<- gsub("IgA1", "skyblue1",
                                 Algorithm_Iso_12345_V)
Algorithm_Iso_12345_V_ann<- gsub("IgA2", "royalblue1",
                                 Algorithm_Iso_12345_V_ann)
Algorithm_Iso_12345_V_ann<- gsub("IgG3", "yellowgreen",
                                 Algorithm_Iso_12345_V_ann)
Algorithm_Iso_12345_V_ann<- gsub("IgG1", "darkgreen",
                                 Algorithm_Iso_12345_V_ann)
Algorithm_Iso_12345_V_ann<- gsub("IgG2", "darkolivegreen1",
                                 Algorithm_Iso_12345_V_ann)
Algorithm_Iso_12345_V_ann<- gsub("IgG Family", "green",
                                 Algorithm_Iso_12345_V_ann)
Algorithm_Iso_12345_V_ann<- gsub("IgA Family", "blue",
                                 Algorithm_Iso_12345_V_ann)
Algorithm_Iso_12345_V_ann<- gsub("not available", "grey",
                                 Algorithm_Iso_12345_V_ann)
Algorithm_Iso_12345_V_ann<- gsub("unclear", "grey",
                                 Algorithm_Iso_12345_V_ann)
```

```
Specificity_Iso_12345_V <- Vac_annotation_known$Specificity
summary(Specificity_Iso_12345_V)
```

```
## negative  vaccine
##      100      195
```

```
Specificity_Iso_12345_V_ann<- gsub("negative", "black",
                                   Specificity_Iso_12345_V)
Specificity_Iso_12345_V_ann<- gsub("vaccine", "blue",
                                   Specificity_Iso_12345_V_ann)
```

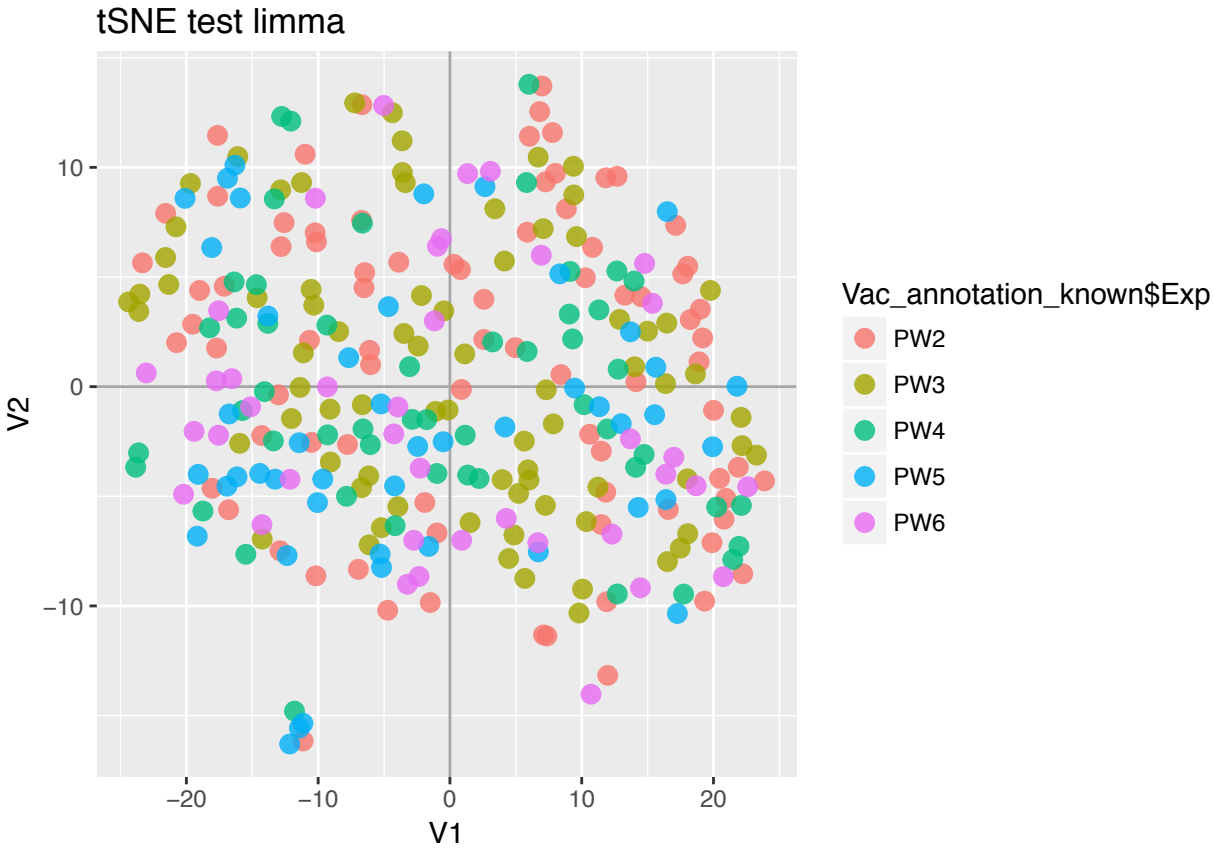
```
I.S.A_12345_V_ann <- cbind(Flow_Iso_12345_V_ann, Sanger_Iso_12345_V_ann,
                           Algorithm_Iso_12345_V_ann,
                           Specificity_Iso_12345_V_ann)
col_I.S.A_12345_V_ann <- as.matrix(I.S.A_12345_V_ann, ncol = 4)
colnames(col_I.S.A_12345_V_ann) <- c("Flow Isotype", "Sanger Seq Isotype",
                                     "BASIC Isotype", "mAb Specificity")
```

```
#heatmap.3(Vaccine_IGHC_genes,
#main="QN Vaccine_IGHC_genes LIMMA b1=Exp b2=Patient d=0+S+I",
#       ColSideColors=col_I.S.A_12345_V_ann, dendrogram = 'column',
#       keysize =.6,col=rev(col_spectrum), scale = 'column')
#       , margins = c(5,9)
#legend(x = 'left', c("IgA family", "IgA1", "IgA2", "IgG family", "IgG1",
#       "IgG3", "IgG2", "unclear", "NA", "vaccine +", "vaccine -",
#       "unknown/failure"), col=c("blue", "skyblue1", "royalblue1","green",
#       "darkgreen", "yellowgreen", "darkolivegreen1", "grey", "grey", "purple",
#       "black", "grey"), lty=1, lwd = 5, cex=0.8, inset = 0, border="black")
```

5.2 3 population tSNE projection

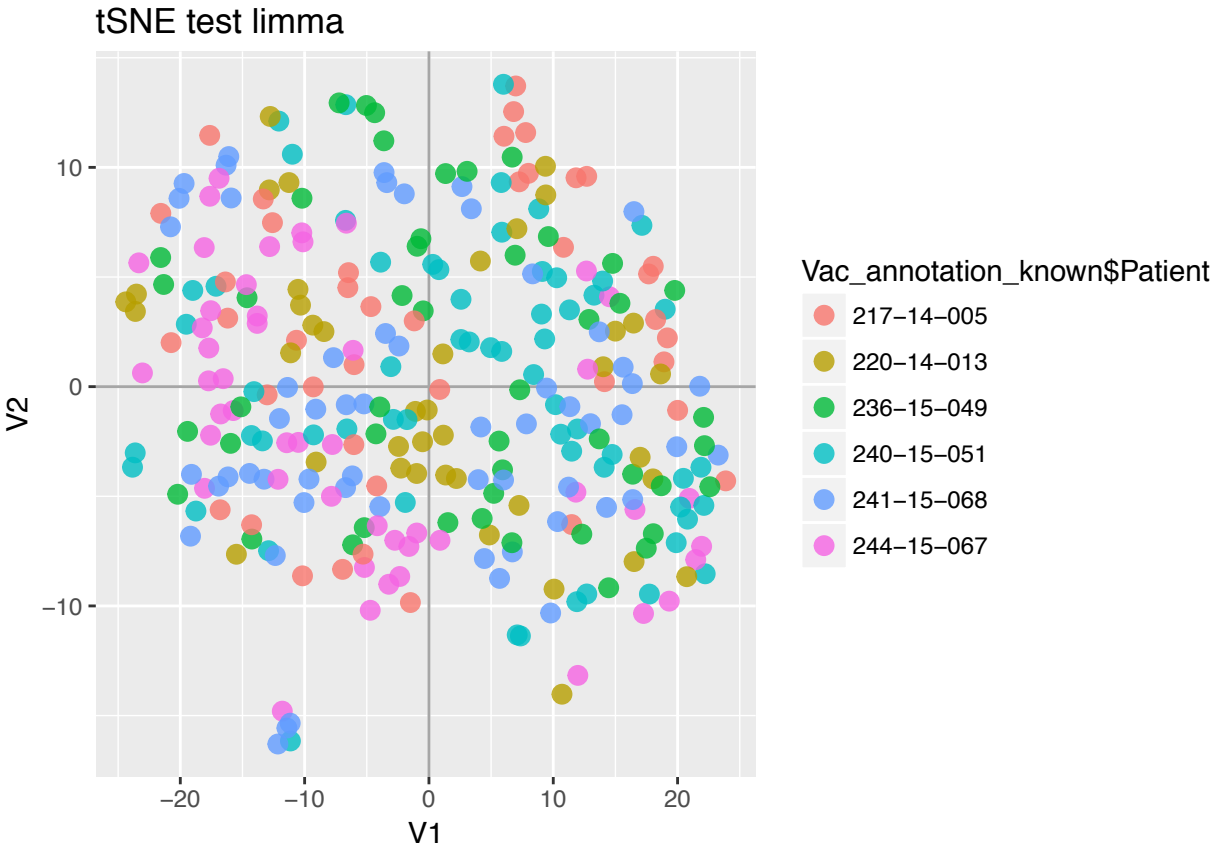
5.2.1 1st - show Experimental batch effect has been removed

```
library(ggplot2)
library(RColorBrewer)
library(colorspace)

set.seed(27)
library(Rtsne)

trans_limmaAdj <- as.data.frame(t(Limma_adj_4))
tSNE_limmaAdj <- Rtsne(data.frame(trans_limmaAdj),
                       check_duplicates = FALSE, theta = 0.001,
                       perplexity = 30, initial_dims = 10)

ID <- ggplot(data = as.data.frame(tSNE_limmaAdj$Y),
             aes(x=V1, y=V2, label = colnames(Limma_adj_4)))  +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vac_annotation_known$Exp),
                   alpha = 0.8, size = 3) +
        ggtitle("tSNE test limma")
ID
```



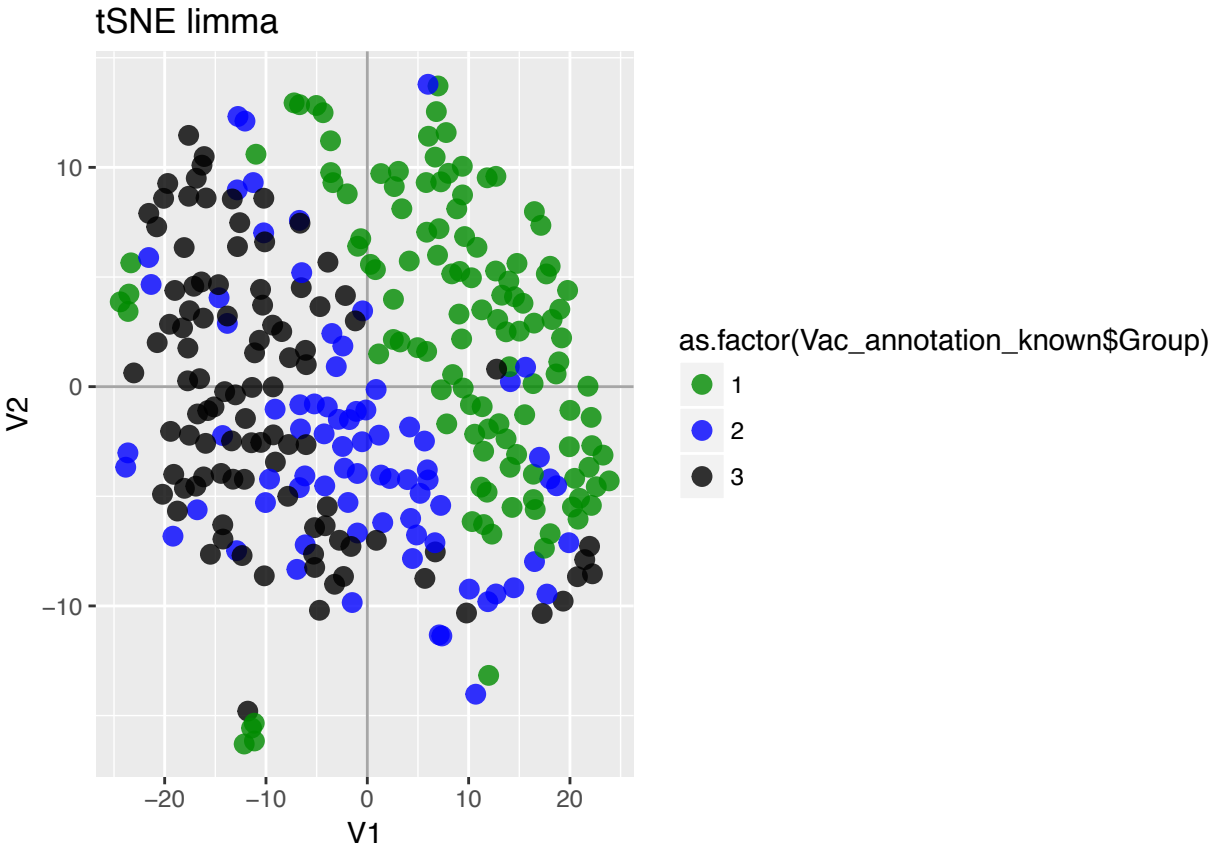
5.2.2 2nd - show Patient Batch effect is removed

```
ID <- ggplot(data = as.data.frame(tSNE_limmaAdj$Y),
             aes(x=V1, y=V2,label = colnames(Limma_adj_4)))  +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vac_annotation_known$Patient),
                   alpha = 0.8, size = 3) +
        ggtitle("tSNE test limma")
ID
```



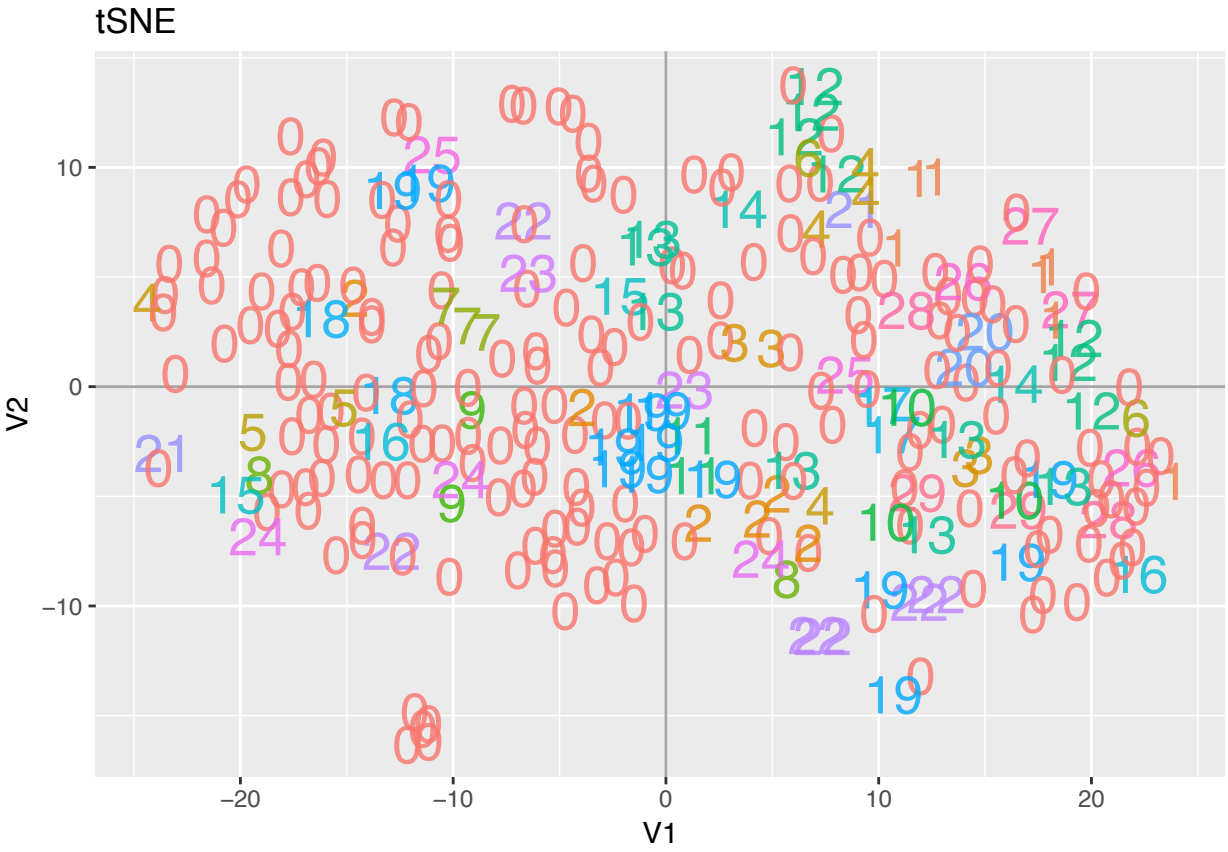
5.2.3 3rd - Look for clustering by group ID

```
ID <- ggplot(data = as.data.frame(tSNE_limmaAdj$Y),
             aes(x=V1, y=V2,label = colnames(Limma_adj_4)))  +
        scale_color_manual(values = c('green4','blue', 'black'))+
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=as.factor(Vac_annotation_known$Group)),
                   alpha = 0.8, size = 3) +
        ggtitle("tSNE limma")
ID
```



5.2.4 4th - color by clone ID

```
Vac_repertoire_known$Clone[is.na(Vac_repertoire_known$Clone)] <- 0
ID <- ggplot(data = as.data.frame(tSNE_limmaAdj$Y),
             aes(x=V1, y=V2, label = colnames(Limma_adj_4))) +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_text(aes(colour=as.factor(Vac_repertoire_known$Clone),
                      label = Vac_repertoire_known$Clone),
                  alpha = 0.8, size = 7) +
        ggtitle("tSNE") + theme(legend.position="none")
ID
```



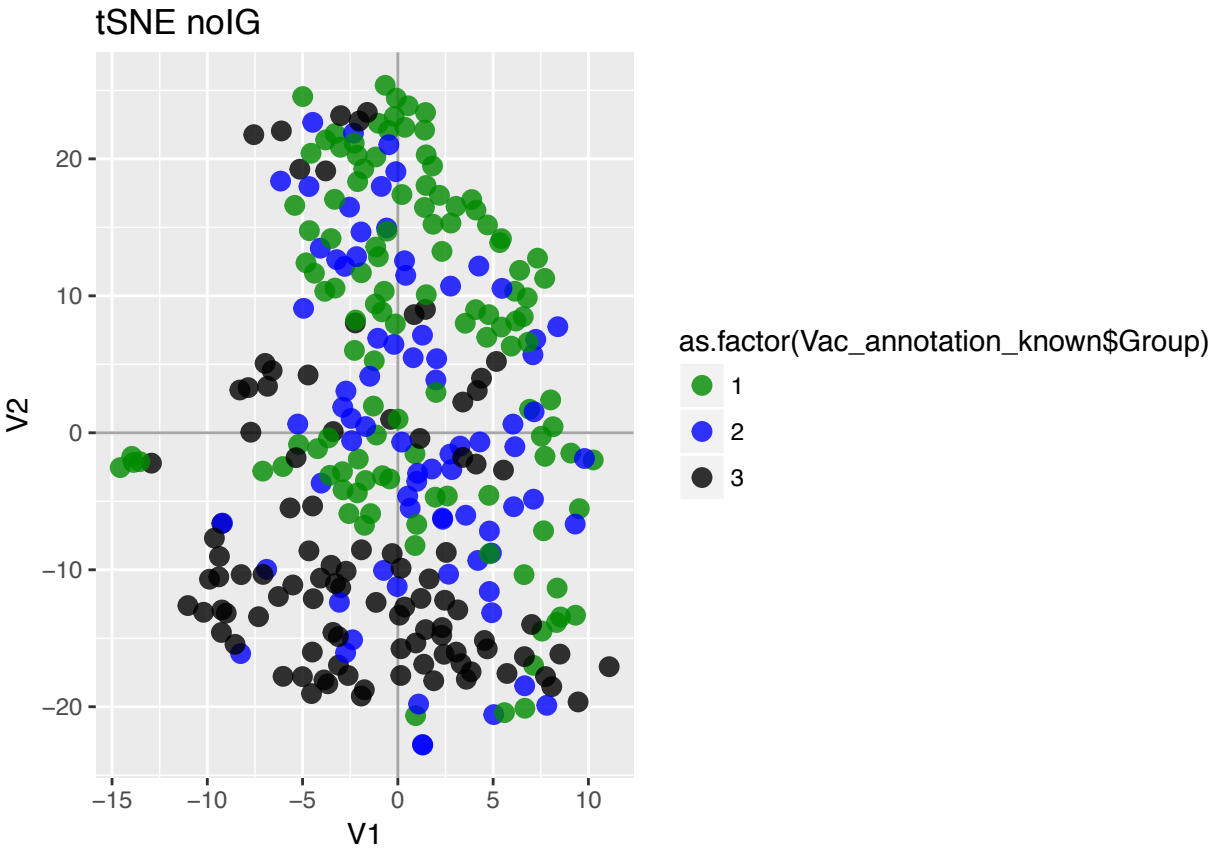
5.2.5 5th- examine how the immunoglobulin genes effect the group clustering. Remove IG before tSNE.

```
###Make the Kallisto IG gene list
IGHSubset <- TagPL12345_genes[grep("^IGH", rownames(TagPL12345_genes)),]
IGKSubset <- TagPL12345_genes[grep("^IGK", rownames(TagPL12345_genes)),]
IGLSubset <- TagPL12345_genes[grep("^IGL", rownames(TagPL12345_genes)),]
kallisto_IGgenes <- rbind(IGHSubset, IGKSubset, IGLSubset)
Kallisto_IG_gene_list <- rownames(kallisto_IGgenes)
Kallisto_IG_gene_list <- c(Kallisto_IG_gene_list, 'AC233755.1')
#remove from Limma_adj_4
limma_adj_noIG <- Limma_adj_4[!rownames(Limma_adj_4)%in%
                                      Kallisto_IG_gene_list,]
dim(limma_adj_noIG)
```

```
## [1] 11687   295
```

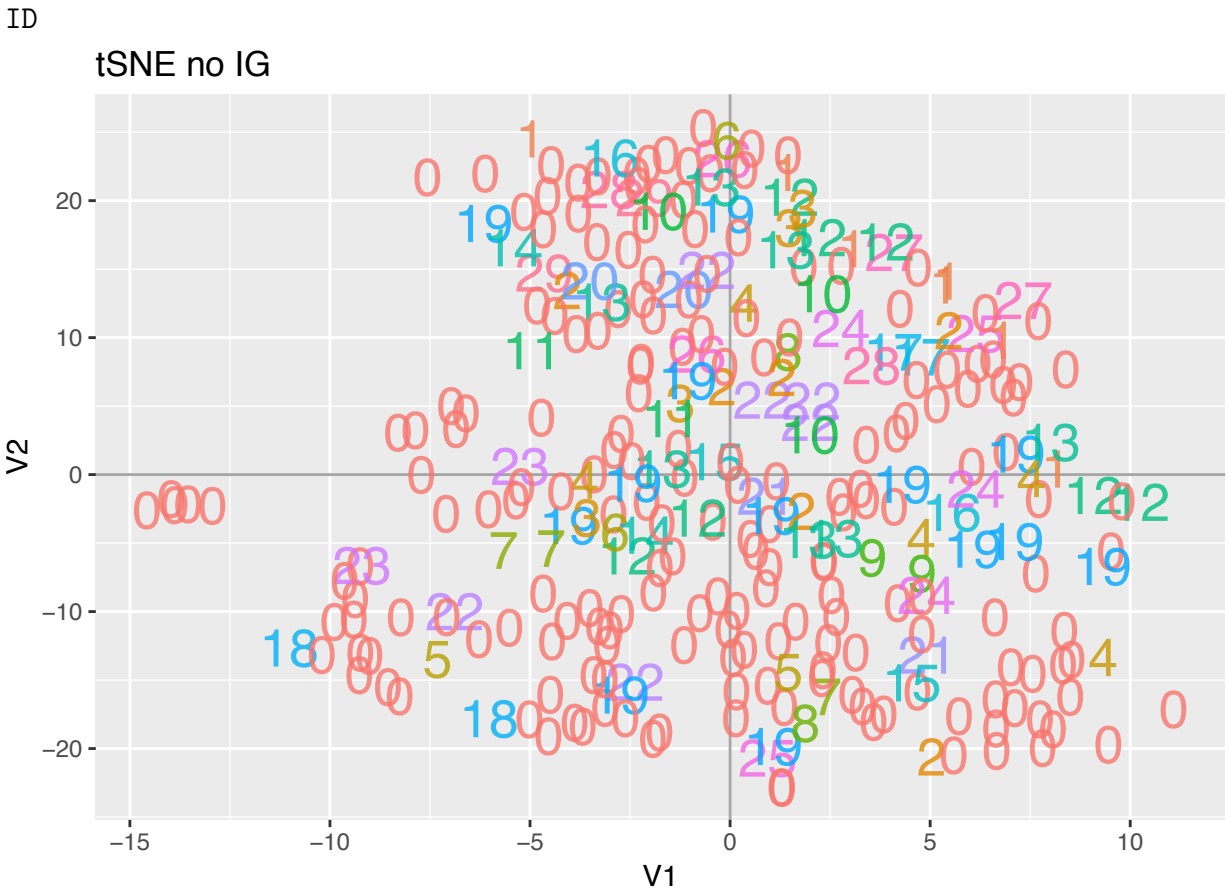
```
#tSNE
trans_limmaAdj_noIG <- as.data.frame(t(limma_adj_noIG))
tSNE_limmaAdj_noIG <- Rtsne(data.frame(trans_limmaAdj_noIG),
                            check_duplicates = FALSE, theta = 0.001,
                            perplexity = 30, initial_dims = 10)
```

```
#tSNE plot
ID <- ggplot(data = as.data.frame(tSNE_limmaAdj_noIG$Y),
             aes(x=V1, y=V2,label = colnames(limma_adj_noIG)))  +
        scale_color_manual(values = c('green4','blue', 'black'))+

        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=as.factor(Vac_annotation_known$Group)),
                   alpha = 0.8, size = 3) +
        ggtitle("tSNE noIG")
ID
```



5.2.6 6th - do the same thing with the clones

```
ID <- ggplot(data = as.data.frame(tSNE_limmaAdj_noIG$Y),
             aes(x=V1, y=V2,label = colnames(Limma_adj_4))) +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_text(aes(colour=as.factor(Vac_repertoire_known$Clone),
                      label = Vac_repertoire_known$Clone),
                  alpha = 0.8, size = 7) +
        ggtitle("tSNE no IG") + theme(legend.position="none")
```



6 Now perform exploration of clones

6.1 Make Clonal DF

```
Clones <- subset(Vac_repertoire_known, Clone != '0')
dim(Clones)
```

```
## [1] 100  41
```

```
#13 cells in Negative clones means 87 in positive
dim(Vac_annotation_known)
```

```
## [1] 295   9
```

```
Annotation_Clones <- Vac_annotation_known[rownames(
        Vac_annotation_known)%in%rownames(Clones),]
dim(Annotation_Clones)
```

```
## [1] 100   9
```

```
all(rownames(Annotation_Clones)==rownames(Clones))
```

```
## [1] TRUE
```

```
Annotation_Clones <- cbind(Annotation_Clones, Clones$Clone)

Clone_DF <- Limma_adj_4[,colnames(Limma_adj_4)
                        %in% rownames(Annotation_Clones)]
dim(Clone_DF)
```

```
## [1] 11895   100
```

6.2 Make not clonal DF

```
Not_Clones <- subset(Vac_repertoire_known, Clone == 0)
dim(Not_Clones)
```

```
## [1] 195  41
```

```
no_clone_DF <- Limma_adj_4[
        ,colnames(Limma_adj_4) %in% rownames(Not_Clones)]
dim(no_clone_DF)
```

```
## [1] 11895   195
```

```
Not_Clones_ANN <- Vac_annotation_known[
        rownames(Vac_annotation_known)%in%
                colnames(no_clone_DF),]
dim(Not_Clones_ANN)
```

```
## [1] 195   9
```

6.3 Make VP and VN Not clones

```
Vaccine_Positive <- subset(Vac_annotation_known, Specificity == 'vaccine')
dim(Vaccine_Positive)
```

```
## [1] 195   9
```

```
Vaccine_pos_Not_Clones <- no_clone_DF[,colnames(
        no_clone_DF)%in%rownames(Vaccine_Positive)]
dim(Vaccine_pos_Not_Clones)
```

```
## [1] 11895   108
```

```
Vaccine_Neg_Not_Clones <- no_clone_DF[
        ,!colnames(no_clone_DF)%in%rownames(Vaccine_Positive)]
dim(Vaccine_Neg_Not_Clones)
```

```
## [1] 11895    87
```

6.4 separate out each clonal family and do within clone correlations

```
table(Annotation_Clones$Specificity, Annotation_Clones$`Clones$Clone`)
```

```
##
##             1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
##   negative  0  0  0  0  2  0  3  2  0  0  0  0  0  0  2  2  0  2  0  0  0
##   vaccine   7  6  4  5  0  2  0  0  2  3  2  7  7  2  0  0  2  0 13  2  2
##
##            22 23 24 25 26 27 28 29
##   negative  0  0  0  0  0  0  0  0
##   vaccine   6  2  3  2  2  2  2  2
```

```
first <- subset(Clones, Clone == '1')
first_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(first)]
first_Cor <- pearson_itteration(first_DF)

second <- subset(Clones, Clone == '2')
second_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(second)]
second_Cor <- pearson_itteration(second_DF)

third <- subset(Clones, Clone == '3')
third_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(third)]
third_Cor <- pearson_itteration(third_DF)

fourth <- subset(Clones, Clone == '4')
fourth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(fourth)]
fourth_Cor <- pearson_itteration(fourth_DF)

fifth <- subset(Clones, Clone == '5')
fifth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(fifth)]
fifth_Cor <- pearson_itteration(fifth_DF)

sixth <- subset(Clones, Clone == '6')
sixth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(sixth)]
sixth_Cor <- pearson_itteration(sixth_DF)

seventh <- subset(Clones, Clone == '7')
seventh_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(seventh)]
seventh_Cor <- pearson_itteration(seventh_DF)

eigth <- subset(Clones, Clone == '8')
eigth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(eigth)]
eigth_Cor <- pearson_itteration(eigth_DF)
```

```
ninth <- subset(Clones, Clone == '9')
ninth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(ninth)]
ninth_Cor <- pearson_itteration(ninth_DF)

tenth <- subset(Clones, Clone == '10')
tenth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(tenth)]
tenth_Cor <- pearson_itteration(tenth_DF)

eleventh <- subset(Clones, Clone == '11')
eleventh_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(eleventh)]
eleventh_Cor <- pearson_itteration(eleventh_DF)

twelfth <- subset(Clones, Clone == '12')
twelfth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twelfth)]
twelfth_Cor <- pearson_itteration(twelfth_DF)

thirteenth <- subset(Clones, Clone == '13')
thirteenth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(thirteenth)]
thirteenth_Cor <- pearson_itteration(thirteenth_DF)

fourteenth <- subset(Clones, Clone == '14')
fourteenth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(fourteenth)]
fourteenth_Cor <- pearson_itteration(fourteenth_DF)

fifteenth <- subset(Clones, Clone == '15')
fifteenth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(fifteenth)]
fifteenth_Cor <- pearson_itteration(fifteenth_DF)

sixteenth <- subset(Clones, Clone == '16')
sixteenth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(sixteenth)]
sixteenth_Cor <- pearson_itteration(sixteenth_DF)

seventeenth <- subset(Clones, Clone == '17')
seventeenth_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(seventeenth)]
seventeenth_Cor <- pearson_itteration(seventeenth_DF)

eighteen <- subset(Clones, Clone == '18')
eighteen_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(eighteen)]
eighteen_Cor <- pearson_itteration(eighteen_DF)

nineteen <- subset(Clones, Clone == '19')
nineteen_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(nineteen)]
nineteen_Cor <- pearson_itteration(nineteen_DF)
```

```
twenty <- subset(Clones, Clone == '20')
twenty_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twenty)]
twenty_Cor <- pearson_itteration(twenty_DF)

twentyone <- subset(Clones, Clone == '21')
twentyone_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentyone)]
twentyone_Cor <- pearson_itteration(twentyone_DF)

twentytwo <- subset(Clones, Clone == '22')
twentytwo_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentytwo)]
twentytwo_Cor <- pearson_itteration(twentytwo_DF)

twentythree <- subset(Clones, Clone == '23')
twentythree_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentythree)]
twentythree_Cor <- pearson_itteration(twentythree_DF)

twentyfour <- subset(Clones, Clone == '24')
twentyfour_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentyfour)]
twentyfour_Cor <- pearson_itteration(twentyfour_DF)

twentyfive <- subset(Clones, Clone == '25')
twentyfive_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentyfive)]
twentyfive_Cor <- pearson_itteration(twentyfive_DF)

twentysix <- subset(Clones, Clone == '26')
twentysix_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentysix)]
twentysix_Cor <- pearson_itteration(twentysix_DF)

twentyseven <- subset(Clones, Clone == '27')
twentyseven_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentyseven)]
twentyseven_Cor <- pearson_itteration(twentyseven_DF)

twentyeight <- subset(Clones, Clone == '28')
twentyeight_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentyeight)]
twentyeight_Cor <- pearson_itteration(twentyeight_DF)

twentynine <- subset(Clones, Clone == '29')
twentynine_DF <- Clone_DF[,colnames(Clone_DF) %in% rownames(twentynine)]
twentynine_Cor <- pearson_itteration(twentynine_DF)
```

6.5 Group together VP within clonal family correlations and VN within clonal family correlations

```
#VP <- 1,2,3,4,6,9,10,11,12,13,14,17,19,20,21,22,23,24,25,26,27,28,29
#VN <- 5,7,8,15,16,18
#G&A clones <- 4, 13, 21

VP_clone_correlations <- c(as.numeric(first_Cor),as.numeric(second_Cor),
                           as.numeric(third_Cor), as.numeric(fourth_Cor),
                           as.numeric(sixth_Cor), as.numeric(ninth_Cor),
                           as.numeric(tenth_Cor), as.numeric(eleventh_Cor),
                           as.numeric(twelfth_Cor), as.numeric(thirteenth_Cor),
                           as.numeric(fourteenth_Cor),
                           as.numeric(seventeenth_Cor),
                           as.numeric(nineteen_Cor),
                           as.numeric(twenty_Cor),
                           as.numeric(twentyone_Cor),
                           as.numeric(twentytwo_Cor),
                           as.numeric(twentythree_Cor),
                           as.numeric(twentyfour_Cor),
                           as.numeric(twentyfive_Cor),
                           as.numeric(twentysix_Cor),
                           as.numeric(twentyseven_Cor),
                           as.numeric(twentyeight_Cor),
                           as.numeric(twentynine_Cor))

VN_clone_correlations <- c(as.numeric(fifth_Cor), as.numeric(seventh_Cor),
                           as.numeric(eigth_Cor), as.numeric(fifteenth_Cor),
                           as.numeric(sixteenth_Cor), as.numeric(eighteen_Cor))
```

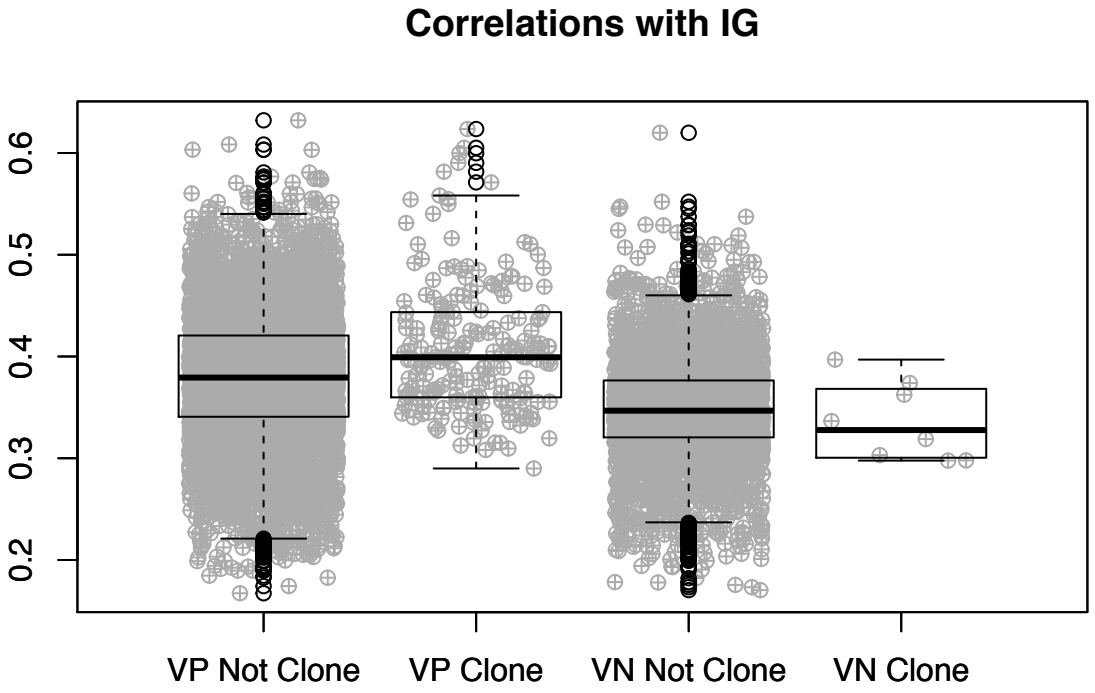
6.6 Boxplot VP/VN clones vs not clones

```
VP_notClones_Cor <- pearson_itteration(Vaccine_pos_Not_Clones)
VN_notClones_Cor <- pearson_itteration(Vaccine_Neg_Not_Clones)
length(VN_clone_correlations)
```

```
## [1] 8
```

```
par(mfrow = c(1,1))

x <- list("VP Not Clone"=as.numeric(VP_notClones_Cor),
          "VP Clone"=VP_clone_correlations,
          "VN Not Clone" = as.numeric(VN_notClones_Cor),
          "VN Clone" = VN_clone_correlations)
```

```
stripchart(x, vertical = TRUE, method = "jitter",
           jitter=.35, pch = 10, col = 'grey67')
boxplot(x, col='transparent', border = 'black',
        alpha=.8, add = TRUE, main = "Correlations with IG")
```



6.7 Same on same data with no IG

```
Vaccine_pos_Not_Clones_noIG <- Vaccine_pos_Not_Clones[!rownames(
        Vaccine_pos_Not_Clones)%in%Kallisto_IG_gene_list,]
Vaccine_Neg_Not_Clones_noIG <- Vaccine_Neg_Not_Clones[!rownames(
        Vaccine_Neg_Not_Clones)%in%Kallisto_IG_gene_list,]
Clones_noIG <- Clone_DF[!rownames(
        Clone_DF)%in%Kallisto_IG_gene_list,]
dim(Clones_noIG)
```

```
## [1] 11687   100
```

6.8 Separate out each clonal group again, this time after IG genes are excluded

```
first <- subset(Clones, Clone == '1')
first_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(first)]
first_Cor <- pearson_itteration(first_DF)
```

```
second <- subset(Clones, Clone == '2')
second_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(second)]
second_Cor <- pearson_itteration(second_DF)

third <- subset(Clones, Clone == '3')
third_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(third)]
third_Cor <- pearson_itteration(third_DF)

fourth <- subset(Clones, Clone == '4')
fourth_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(fourth)]
fourth_Cor <- pearson_itteration(fourth_DF)

fifth <- subset(Clones, Clone == '5')
fifth_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(fifth)]
fifth_Cor <- pearson_itteration(fifth_DF)

sixth <- subset(Clones, Clone == '6')
sixth_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(sixth)]
sixth_Cor <- pearson_itteration(sixth_DF)

seventh <- subset(Clones, Clone == '7')
seventh_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(seventh)]
seventh_Cor <- pearson_itteration(seventh_DF)

eigth <- subset(Clones, Clone == '8')
eigth_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(eigth)]
eigth_Cor <- pearson_itteration(eigth_DF)

ninth <- subset(Clones, Clone == '9')
ninth_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(ninth)]
ninth_Cor <- pearson_itteration(ninth_DF)

tenth <- subset(Clones, Clone == '10')
tenth_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(tenth)]
tenth_Cor <- pearson_itteration(tenth_DF)

eleventh <- subset(Clones, Clone == '11')
eleventh_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(eleventh)]
eleventh_Cor <- pearson_itteration(eleventh_DF)

twelfth <- subset(Clones, Clone == '12')
twelfth_DF <- Clones_noIG[,colnames(Clones_noIG) %in% rownames(twelfth)]
twelfth_Cor <- pearson_itteration(twelfth_DF)
```

```
thirteenth <- subset(Clones, Clone == '13')
thirteenth_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(thirteenth)]
thirteenth_Cor <- pearson_itteration(thirteenth_DF)

fourteenth <- subset(Clones, Clone == '14')
fourteenth_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(fourteenth)]
fourteenth_Cor <- pearson_itteration(fourteenth_DF)

fifteenth <- subset(Clones, Clone == '15')
fifteenth_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(fifteenth)]
fifteenth_Cor <- pearson_itteration(fifteenth_DF)

sixteenth <- subset(Clones, Clone == '16')
sixteenth_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(sixteenth)]
sixteenth_Cor <- pearson_itteration(sixteenth_DF)

seventeenth <- subset(Clones, Clone == '17')
seventeenth_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(seventeenth)]
seventeenth_Cor <- pearson_itteration(seventeenth_DF)

eighteen <- subset(Clones, Clone == '18')
eighteen_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(eighteen)]
eighteen_Cor <- pearson_itteration(eighteen_DF)

nineteen <- subset(Clones, Clone == '19')
nineteen_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(nineteen)]
nineteen_Cor <- pearson_itteration(nineteen_DF)

twenty <- subset(Clones, Clone == '20')
twenty_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twenty)]
twenty_Cor <- pearson_itteration(twenty_DF)

twentyone <- subset(Clones, Clone == '21')
twentyone_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentyone)]
twentyone_Cor <- pearson_itteration(twentyone_DF)
```

```
twentytwo <- subset(Clones, Clone == '22')
twentytwo_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentytwo)]
twentytwo_Cor <- pearson_itteration(twentytwo_DF)

twentythree <- subset(Clones, Clone == '23')
twentythree_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentythree)]
twentythree_Cor <- pearson_itteration(twentythree_DF)

twentyfour <- subset(Clones, Clone == '24')
twentyfour_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentyfour)]
twentyfour_Cor <- pearson_itteration(twentyfour_DF)

twentyfive <- subset(Clones, Clone == '25')
twentyfive_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentyfive)]
twentyfive_Cor <- pearson_itteration(twentyfive_DF)

twentysix <- subset(Clones, Clone == '26')
twentysix_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentysix)]
twentysix_Cor <- pearson_itteration(twentysix_DF)

twentyseven <- subset(Clones, Clone == '27')
twentyseven_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentyseven)]
twentyseven_Cor <- pearson_itteration(twentyseven_DF)

twentyeight <- subset(Clones, Clone == '28')
twentyeight_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentyeight)]
twentyeight_Cor <- pearson_itteration(twentyeight_DF)

twentynine <- subset(Clones, Clone == '29')
twentynine_DF <- Clones_noIG[
        ,colnames(Clones_noIG) %in% rownames(twentynine)]
twentynine_Cor <- pearson_itteration(twentynine_DF)
```

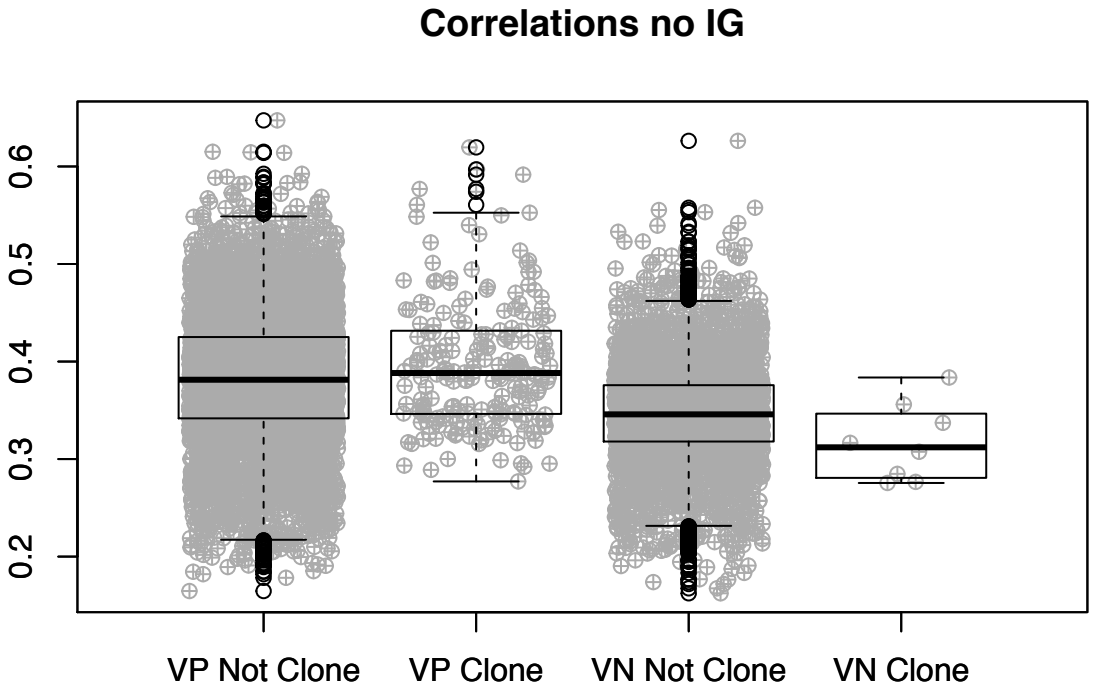
6.9 Group no-IG clones together; VP within clonal family correlations and VN within clonal family correlations

```
VP_clone_correlations_noIG <- c(as.numeric(first_Cor),as.numeric(second_Cor),
                                as.numeric(third_Cor), as.numeric(fourth_Cor),
                                as.numeric(sixth_Cor), as.numeric(ninth_Cor),
                                as.numeric(tenth_Cor),
                                as.numeric(eleventh_Cor),
                                as.numeric(twelfth_Cor),
                                as.numeric(thirteenth_Cor),
                                as.numeric(fourteenth_Cor),
                                as.numeric(seventeenth_Cor),
                                as.numeric(nineteen_Cor),
                                as.numeric(twenty_Cor),
                                as.numeric(twentyone_Cor),
                                as.numeric(twentytwo_Cor),
                                as.numeric(twentythree_Cor),
                                as.numeric(twentyfour_Cor),
                                as.numeric(twentyfive_Cor),
                                as.numeric(twentysix_Cor),
                                as.numeric(twentyseven_Cor),
                                as.numeric(twentyeight_Cor),
                                as.numeric(twentynine_Cor))

VN_clone_correlations_noIG <- c(as.numeric(fifth_Cor),as.numeric(seventh_Cor),
                                as.numeric(eigth_Cor),
                                as.numeric(fifteenth_Cor),
                                as.numeric(sixteenth_Cor),
                                as.numeric(eighteen_Cor))
```

6.10 Boxplot VP/VN no Ig clones vs not clones

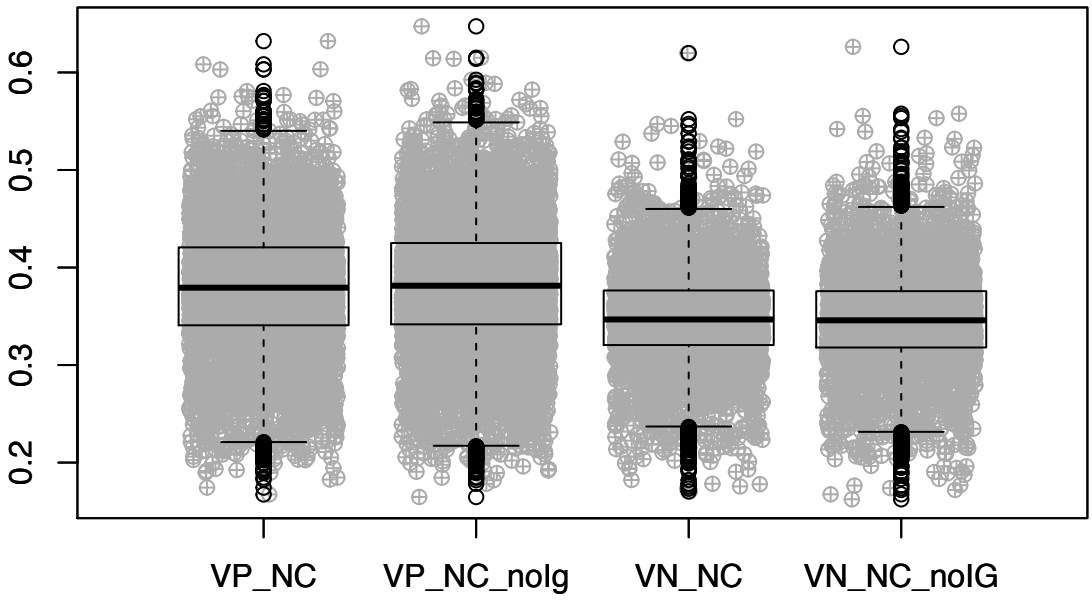
```
Vaccine_Neg_Not_Clones_noIG_COR <- pearson_itteration(
        Vaccine_Neg_Not_Clones_noIG)
Vaccine_pos_Not_Clones_noIG_COR <- pearson_itteration(
        Vaccine_pos_Not_Clones_noIG)

par(mfrow = c(1,1))
x_noIG <- list("VP Not Clone"=as.numeric(Vaccine_pos_Not_Clones_noIG_COR),
               "VP Clone"=VP_clone_correlations_noIG,
               "VN Not Clone" = as.numeric(Vaccine_Neg_Not_Clones_noIG_COR),
               "VN Clone" = VN_clone_correlations_noIG)
```

```
stripchart(x_noIG, vertical = TRUE, method = "jitter", jitter=.35,
           pch = 10, col = 'grey67')
boxplot(x_noIG, col='transparent', border = 'black', alpha=.8, add = TRUE,
        main = "Correlations no IG")
```

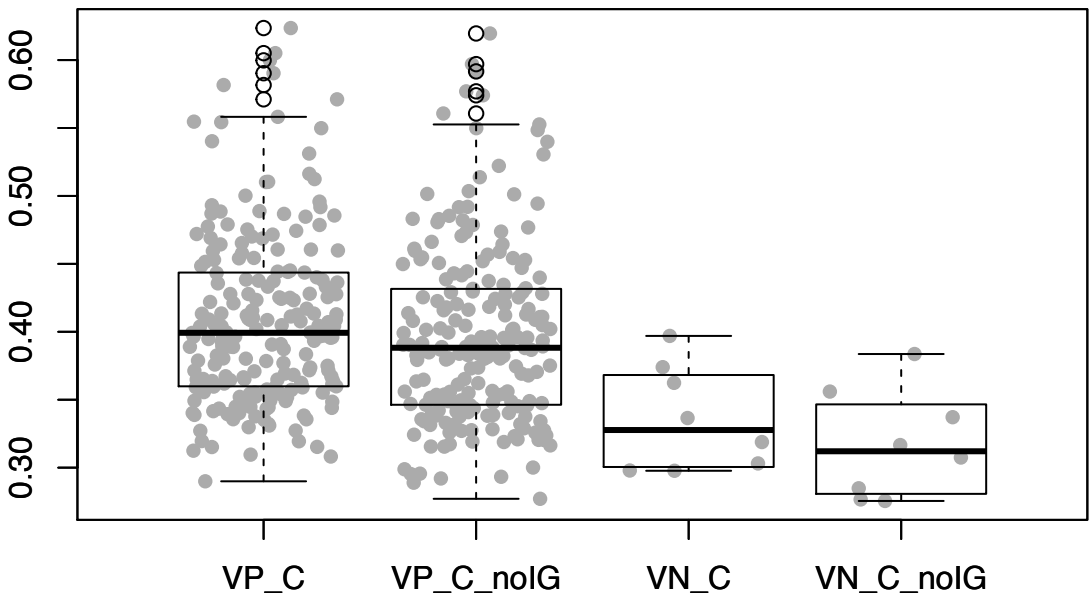


6.11 Boxplot of not clones with and without IG & Clones with and without IG

```
NC <- list('VP_NC' = as.numeric(VP_notClones_Cor),
           'VP_NC_noIg' = as.numeric(Vaccine_pos_Not_Clones_noIG_COR),
           'VN_NC' = as.numeric(VN_notClones_Cor),
           'VN_NC_noIG' = as.numeric(Vaccine_Neg_Not_Clones_noIG_COR))

stripchart(NC, vertical = TRUE, method = "jitter", jitter=.35,
           pch = 10, col = 'grey67')
boxplot(NC, col='transparent', border = 'black', alpha=.8, add = TRUE,
        main = "Not clones with/without Ig")
```




```
C<- list('VP_C' = VP_clone_correlations,
         'VP_C_noIG' = VP_clone_correlations_noIG,
         'VN_C'=VN_clone_correlations,
         'VN_C_noIG'=VN_clone_correlations_noIG)
stripchart(C, vertical = TRUE, method = "jitter", jitter=.35,
           pch = 16, col = 'grey67')
boxplot(C, col='transparent', border = 'black', alpha=.8, add = TRUE,
        main = "Clones with/without IG")
```



7 Repertoire analysis of all three populations - excluding repeate BCRs that are a part of clonal expansions as this will skew gene usage frequency.

```
Repertoire_noExpansion <- subset(Vac_repertoire_known,
                                 Rep_exclude != 'exclude')
dim(Repertoire_noExpansion)
```

```
## [1] 224  41
```

```
No_Expansion_Annotation <- Vac_annotation_known[rownames(
        Vac_annotation_known)%in%rownames(Repertoire_noExpansion),]
dim(No_Expansion_Annotation)
```

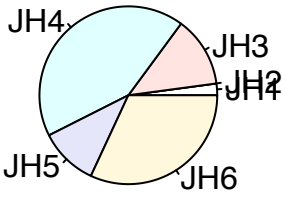
```
## [1] 224   9
```

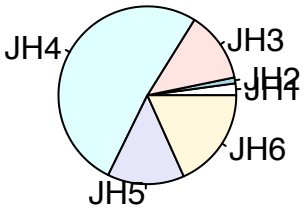
```
IgAP_no_Exp_ann <- subset(No_Expansion_Annotation, Isotype == 'IgA' &
                                 Specificity == 'vaccine')
IgAN_no_Exp_ann <- subset(No_Expansion_Annotation, Isotype == 'IgA' &
                                 Specificity == 'negative')
IgGP_no_Exp_ann <- subset(No_Expansion_Annotation, Isotype == 'IgG' &
                                 Specificity == 'vaccine')

IgAP_rep <- Repertoire_noExpansion[
        rownames(Repertoire_noExpansion)
        %in% rownames(IgAP_no_Exp_ann),]
IgAN_rep <- Repertoire_noExpansion[
        rownames(Repertoire_noExpansion)
        %in% rownames(IgAN_no_Exp_ann),]
IgG_rep <- Repertoire_noExpansion[
        rownames(Repertoire_noExpansion)
        %in% rownames(IgGP_no_Exp_ann),]
#Dont DropLevels or else they will not be c-bind able
```

7.1 Make repertoire Figures

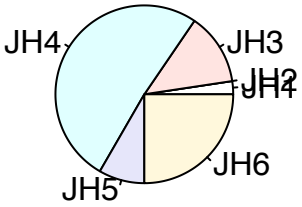
```
par(mfrow=c(1,1), mar = c(6,5,5,5))
#HC J gene usage
JH_locus_by_Isotype <- as.data.frame(cbind
                                     (IgA_J = table(IgAP_rep$JH.locus),
                                           IgAN_J = table(IgAN_rep$JH.locus),
                                           IgG_J = table(IgG_rep$JH.locus)))
#Labeled with gene
par(mfrow=c(1,2))
```

```
pie(JH_locus_by_Isotype$IgA_J, labels = rownames(JH_locus_by_Isotype),
    main = 'IgAP J chain')
pie(JH_locus_by_Isotype$IgAN_J, labels = rownames(JH_locus_by_Isotype),
    main = 'IgAN J chain')
```

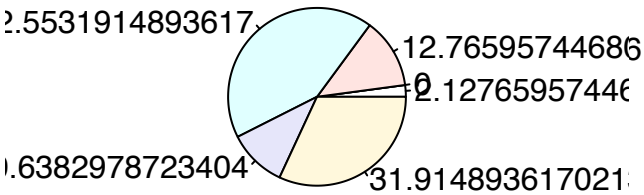


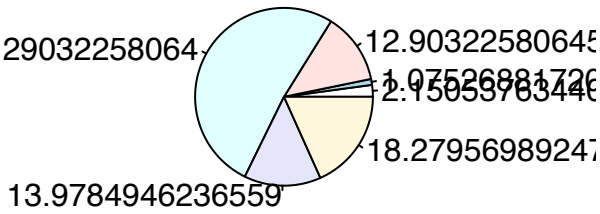



```
pie(JH_locus_by_Isotype$IgG_J, labels = rownames(JH_locus_by_Isotype),
    main = 'IgGP J chain')
#Labeled with frequency
par(mfrow=c(1,2))
```

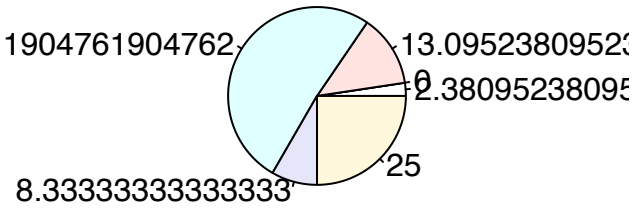



```
pie(JH_locus_by_Isotype$IgA_J, labels =
            prop.table(as.matrix(JH_locus_by_Isotype$IgA_J),2)*100,
    main = 'IgAP J chain')
pie(JH_locus_by_Isotype$IgAN_J, labels =
            prop.table(as.matrix(JH_locus_by_Isotype$IgAN_J),2)*100,
    main = 'IgAN J chain')
```



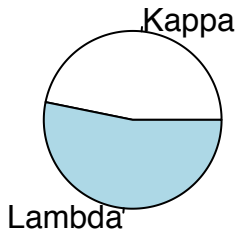



```
pie(JH_locus_by_Isotype$IgG_J, labels =
            prop.table(as.matrix(JH_locus_by_Isotype$IgG_J),2)*100,
    main = 'IgGP J chain')

#Light chain usage
LC_Iso_by_Isotype <- as.data.frame(cbind(IgA_LC = table(IgAP_rep$Isotype),
                                         IgAN_LC = table(IgAN_rep$Isotype),
                                         IgG_LC = table(IgG_rep$Isotype)))

#Labeled with gene
par(mfrow=c(1,2))
```

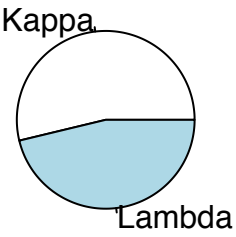



```
pie(LC_Iso_by_Isotype$IgA_LC, labels = rownames(LC_Iso_by_Isotype),
    main = 'IgAP_LC chain')
pie(LC_Iso_by_Isotype$IgAN_LC, labels = rownames(LC_Iso_by_Isotype),
    main = 'IgAN_LC chain')
```

IgAP_LC chain

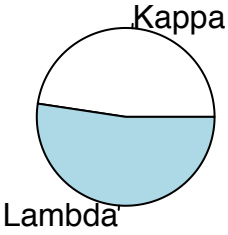


IgAN_LC chain

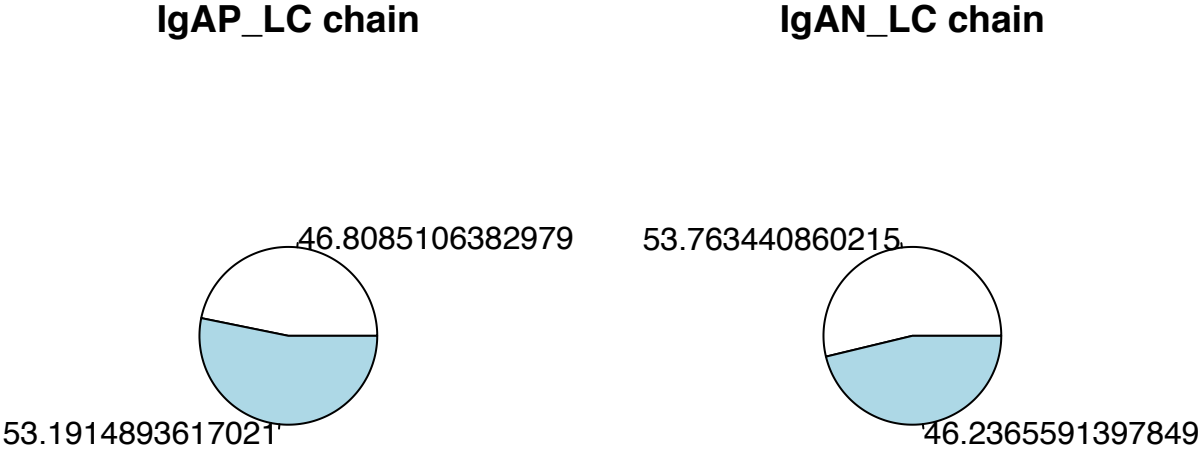



```
pie(LC_Iso_by_Isotype$IgG_LC, labels = rownames(LC_Iso_by_Isotype),
    main = 'IgG_LC chain')
#Labeled with frequency
par(mfrow=c(1,2))
```

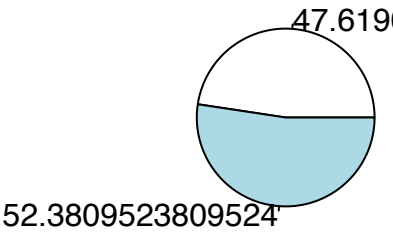
IgG_LC chain



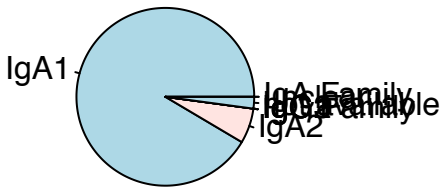

```
pie(LC_Iso_by_Isotype$IgA_LC, labels =
            prop.table(as.matrix(LC_Iso_by_Isotype$IgA_LC))*100,
    main = 'IgAP_LC chain')
pie(LC_Iso_by_Isotype$IgAN_LC, labels =
            prop.table(as.matrix(LC_Iso_by_Isotype$IgAN_LC))*100,
    main = 'IgAN_LC chain')
```

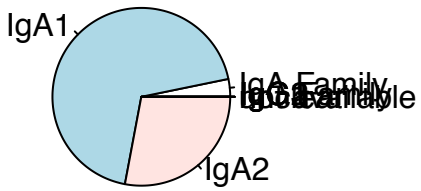



```
pie(LC_Iso_by_Isotype$IgG_LC, labels =
            prop.table(as.matrix(LC_Iso_by_Isotype$IgG_LC))*100,
    main = 'IgG_LC chain')

#HC Isotype
HC_locus_by_Isotype <- as.data.frame(cbind
                                     (IgAP_V = table(IgAP_rep$BASIC.Isotype),
                                           IgAN_V = table(IgAN_rep$BASIC.Isotype),
                                           IgG_V = table(IgG_rep$BASIC.Isotype)))

#Labeled with Isotype
par(mfrow = c(1,2))
```

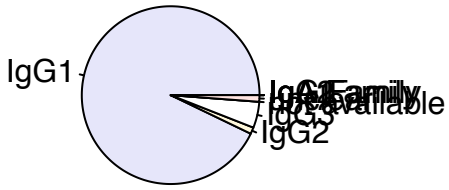



```
pie(HC_locus_by_Isotype$IgAP_V, labels = rownames(HC_locus_by_Isotype),
    main = 'IgAP Iso')
pie(HC_locus_by_Isotype$IgAN_V, labels = rownames(HC_locus_by_Isotype),
    main = 'IgAN Iso')
```

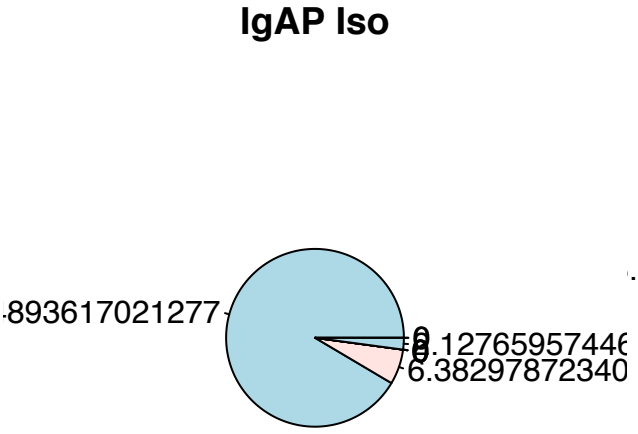


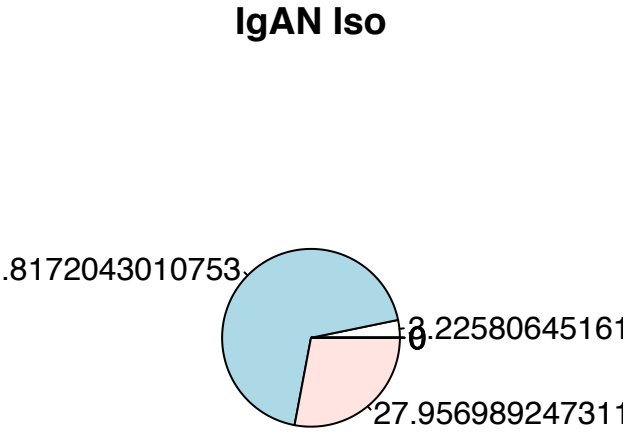



```
pie(HC_locus_by_Isotype$IgG_V, labels = rownames(HC_locus_by_Isotype),
    main = 'IgG Iso')
#Labeled with frequency
par(mfrow = c(1,2))
```

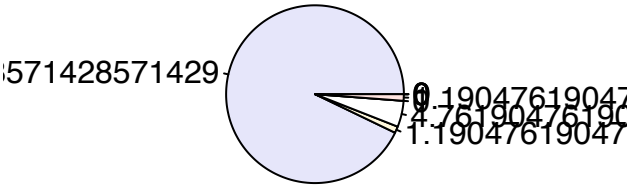



```
pie(HC_locus_by_Isotype$IgAP_V,
    labels = prop.table(as.matrix(HC_locus_by_Isotype$IgAP_V))*100,
    main = 'IgAP Iso')
pie(HC_locus_by_Isotype$IgAN_V,
    labels = prop.table(as.matrix(HC_locus_by_Isotype$IgAN_V))*100,
    main = 'IgAN Iso')
```

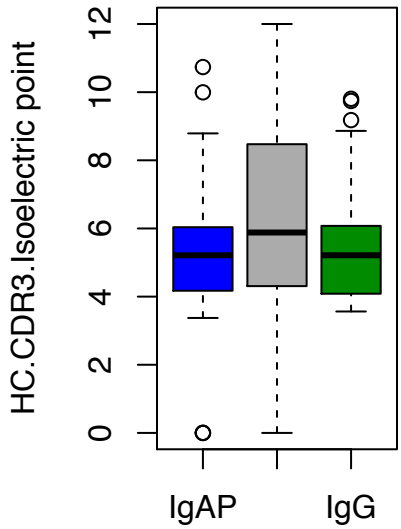


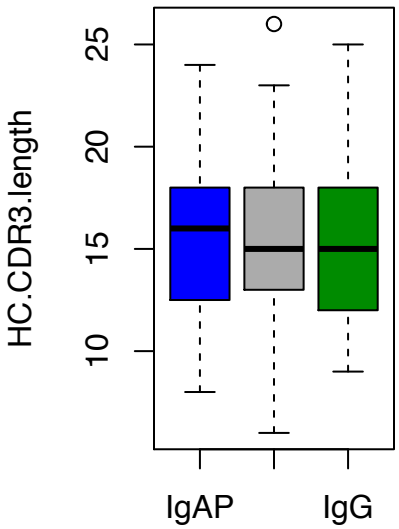



```
pie(HC_locus_by_Isotype$IgG_V,
    labels = prop.table(as.matrix(HC_locus_by_Isotype$IgG_V))*100,
    main = 'IgGP Iso')

#Boxplots for remaining analysis
par(mfrow=c(1,2))
```

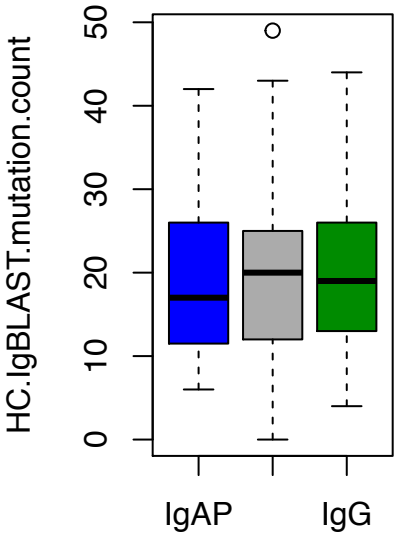


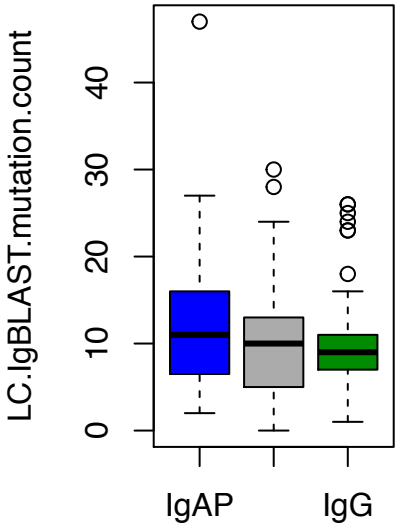

```
#CDR3 Isoelectric
boxplot(IgAP_rep$HC.CDR3.isoelectric.point, IgAN_rep$HC.CDR3.isoelectric.point,
        IgG_rep$HC.CDR3.isoelectric.point, names = c('IgAP', 'IgAN', 'IgG'),
        ylab = 'HC.CDR3.Isoelectric point', col = c('blue', 'grey67','green4'))

#CDR3 length
boxplot(IgAP_rep$HC.CDR3.length, IgAN_rep$HC.CDR3.length,
        IgG_rep$HC.CDR3.length, names = c('IgAP', 'IgAN', 'IgG'),
        ylab = 'HC.CDR3.length', col = c('blue', 'grey67','green4'))
```






```
#HC Ig blast mutation count
boxplot(IgAP_rep$HC.IgBLAST.mutation.count, IgAN_rep$HC.IgBLAST.mutation.count,
        IgG_rep$HC.IgBLAST.mutation.count, names = c('IgAP', 'IgAN', 'IgG'),
        ylab = 'HC.IgBLAST.mutation.count', col = c('blue', 'grey67','green4'))

#LC Ig blast mutation count
boxplot(IgAP_rep$IgBLAST.mutation.count, IgAN_rep$IgBLAST.mutation.count,
        IgG_rep$IgBLAST.mutation.count, names = c('IgAP', 'IgAN', 'IgG'),
        ylab = 'LC.IgBLAST.mutation.count', col = c('blue', 'grey67','green4'))
```

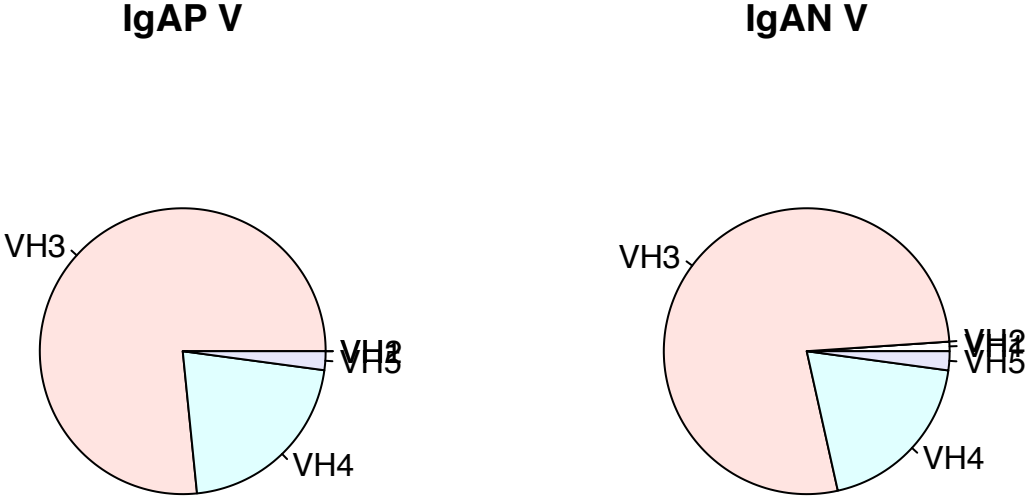




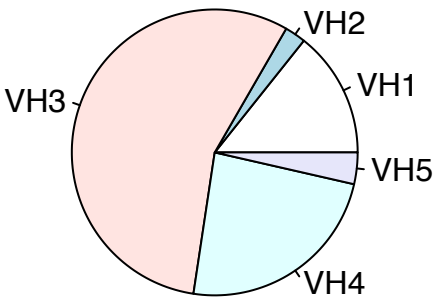
7.2 VH family frequency plots made from an excel table

```
par(mfrow=c(1,2))
pie(as.matrix(V_all_freq$IgAP_V), labels = rownames(V_all_freq),
    main = 'IgAP V')
```

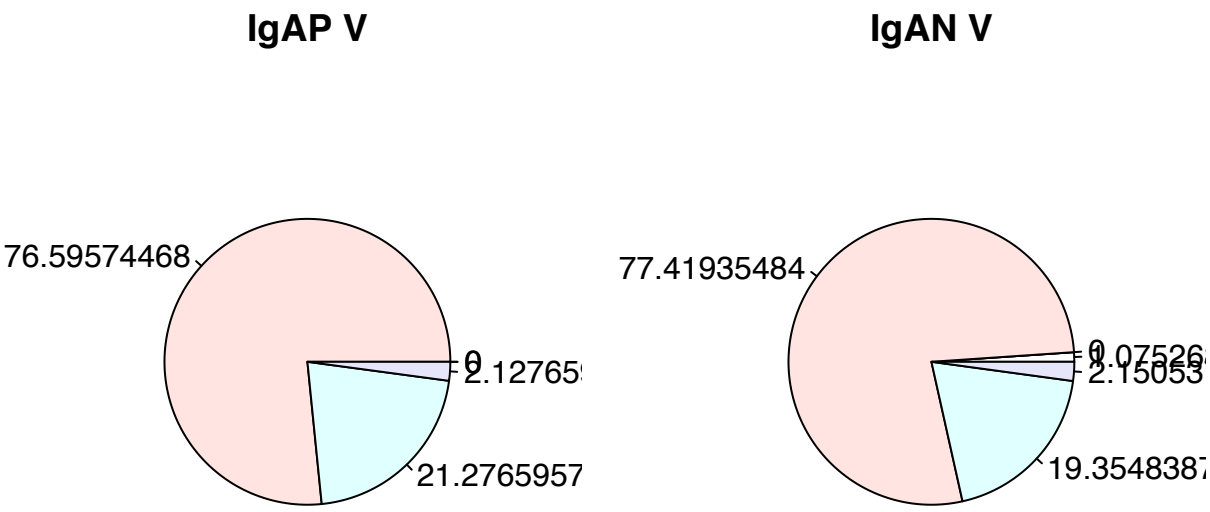
```
pie(as.matrix(V_all_freq$IgAN_V), labels = rownames(V_all_freq),
    main = 'IgAN V')
```



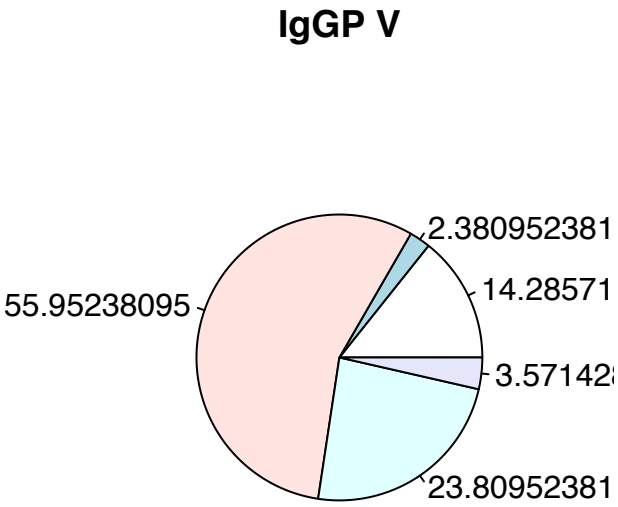

```
pie(as.matrix(V_all_freq$IgG_V), labels =
            rownames(V_all_freq),
    main = 'IgGP V')
#Label by frequency
par(mfrow=c(1,2))
```




```
pie(as.matrix(V_all_freq$IgAP_V), labels = as.matrix(V_all_freq$IgAP_V),
    main = 'IgAP V')
pie(as.matrix(V_all_freq$IgAN_V), labels = as.matrix(V_all_freq$IgAN_V),
    main = 'IgAN V')
```




```
pie(as.matrix(V_all_freq$IgG_V), labels = as.matrix(V_all_freq$IgG_V),
    main = 'IgGP V')
```



8 Here we explore the expression of genes associated with genes from pathways we expect to be different between our populations.

Overall differences between three populations could be due to differential expression of genes indicative of BCR signaling or B cell activation differences between vaccine positive and vaccine negative cells. We also may detect mucosal homing within the IgA vaccine negative population.

8.1 How does BCR signaling look between our three populations?

```
# biocarta_BCRsignaling
BCR_signaling_list <- c("AKT1", "AKT2", "AKT3", "BCL10", "BLNK",
    "BTK", "CARD11", "CD19", "CD22", "CD72", "CD79A", "CD79B",
    "CD81", "CHUK", "CR2", "DAPP1", "FCGR2B", "FOS", "GRB2",
    "GSK3B", "HRAS", "IFITM1", "IGH", "IKBKB", "IKBKG", "INPP5D",
    "INPPL1", "JUN", "KRAS", "LILRB3", "LYN", "MALT1", "MAP2K1",
    "MAP2K2", "MAPK1", "MAPK3", "NFATC1", "NFATC2", "NFATC3",
    "NFKB1", "NFKBIA", "NFKBIB", "NFKBIE", "NRAS", "PIK3AP1",
    "PIK3CA", "PIK3CB", "PIK3CD", "PIK3CG", "PIK3R1", "PIK3R2",
    "PIK3R3", "PIK3R5", "PLCG2", "PPP3CA", "PPP3CB", "PPP3CC",
    "PPP3R1", "PPP3R2", "PRKCB", "PTPN6", "RAC1", "RAC2", "RAC3",
    "RAF1", "RASGRP3", "RELA", "SOS1", "SOS2", "SYK", "VAV1",
    "VAV2", "VAV3")

BCR_sig_list_all <- Limma_adj_4[rownames(Limma_adj_4) %in% BCR_signaling_list,
    ]

Group_All <- Vac_annotation_known$Group
Ann_Group_All <- gsub("1", "green", Group_All)
Ann_Group_All <- gsub("2", "blue", Ann_Group_All)
Ann_Group_All <- gsub("3", "black", Ann_Group_All)

Col_Ann_Group_All <- as.matrix(Ann_Group_All, ncol = 1)

# heatmap.3(BCR_sig_list_all, main='BCR_sig_list_all',
# ColSideColors=Col_Ann_Group_All, margins = c(5,9),keysize
# =.6,col=rev(col_spectrum), scale= 'row')

# legend(x = 'left', c('IgGVP', 'IgAVP', 'IgAVN'),
# col=c('green', 'blue', 'black'), lty=1, lwd = 5, cex=0.8,
# inset = 0, border='black')
```

8.2 What about B cell activation?

```
BCR_act_list <- c("A0A087WW49", "ABL1", "ADA", "ADAM17", "ADGRG3",
    "AHR", "AICDA", "AKAP17A", "APLF", "ATAD5", "ATM", "ATP11C",
    "BAD", "BAK1", "BANK1", "BATF", "BAX", "BCL2", "BCL3", "BCL6",
    "BLNK", "BST1", "BST2", "BTK", "CARD11", "CASP3", "CASP8",
    "CCR6", "CD180", "CD27", "CD28", "CD19", "CD25", "CD30",
    "CD300A", "CD320", "CD38", "CD40", "CD40LG", "CD74", "CD79A",
    "CD79B", "CD81", "CD86", "CDH17", "CDKN1A", "CEBPG", "CHRNA4",
    "CHRNB2", "CLCF1", "CMTM7", "CR2", "CTLA4", "CTPS1", "CXCR5",
```

```
    "CYLD", "DCAF1", "DCLRE1C", "DLL1", "DOCK10", "DOCK11", "EP300",
    "ERCC1", "EXO1", "EXOSC3", "EXOSC6", "FAS", "FLT3", "FNIP1",
    "FOXJ1", "FOXP3", "FZD9", "GAPT", "GON4L", "GPR183", "HDAC4",
    "HDAC5", "HDAC9", "HHEX", "HSPD1", "ICOSLG", "IFNA1", "IFNA10",
    "IFNA14", "IFNA16", "IFNA17", "IFNA2", "IFNA21", "IFNA4",
    "IFNA5", "IFNA6", "IFNA7", "IFNA8", "IFNB1", "IFNE", "IFNG",
    "IFNK", "IFNW1", "IGBP1", "IGHA1", "IGHA2", "IGHD", "IGHE",
    "IGHG1", "IGHG2", "IGHG3", "IGHG4", "IGHM", "IGHV1-18", "IGHV1-24",
    "IGHV1-3", "IGHV1-45", "IGHV1-58", "IGHV1-69-2", "IGHV1-69D",
    "IGHV1OR15-1", "IGHV1OR15-9", "IGHV2-26", "IGHV2-70D", "IGHV2OR16-5",
    "IGHV3-15", "IGHV3-16", "IGHV3-20", "IGHV3-21", "IGHV3-23",
    "IGHV3-35", "IGHV3-38", "IGHV3-43", "IGHV3-49", "IGHV3-64",
    "IGHV3-66", "IGHV3-72", "IGHV3-73", "IGHV3-74", "IGHV3OR15-7",
    "IGHV3OR16-10", "IGHV3OR16-12", "IGHV3OR16-13", "IGHV3OR16-8",
    "IGHV3OR16-9", "IGHV4-28", "IGHV4-30-2", "IGHV4-4", "IGHV4-61",
    "IGHV5-51", "IGHV6-1", "IGHV7-81", "IGKC", "IGLC1", "IGLC6",
    "IGLC7", "IGLL1", "IGLL5", "IKZF3", "IL10", "IL11", "IL13",
    "IL2", "IL21", "IL27RA", "IL4", "IL5", "IL6", "IL7", "IL7R",
    "INHA", "INHBA", "INPP5D", "IRS2", "ITFG2", "ITGA4", "ITGB1",
    "ITM2A", "JAK3", "KIT", "KLF6", "LAT2", "LAX1", "LEF1", "LFNG",
    "LGALS1", "LIG4", "LRRC8A", "LYL1", "LYN", "MALT1", "MEF2C",
    "MFNG", "MIF", "MLH1", "MMP14", "MNDA", "MS4A1", "MSH2",
    "MSH6", "MZB1", "NBN", "NCKAP1L", "NDFIP1", "NFAM1", "NFATC2",
    "NHEJ1", "NKX2-3", "NOD2", "NOTCH2", "NSD2", "NTRK1", "ONECUT1",
    "PAWR", "PAXIP1", "PCID2", "PELI1", "PIK3CD", "PIK3R1", "PKN1",
    "PLCG2", "PLCL2", "POLM", "POU1F1", "POU2F2", "PPP2R3C",
    "PRDM1", "PRKCB", "PRKCD", "PRKDC", "PTK2B", "PTPN2", "PTPN6",
    "PTPRC", "RAG1", "RAG2", "RBPJ", "RC3H1", "RIF1", "RNF168",
    "RNF8", "S4R3C0", "SAMSN1", "SASH3", "SFRP1", "SHB", "SKAP2",
    "SLA2", "SLC39A10", "SP3", "STAT5B", "STAT6", "SUPT6H", "SWAP70",
    "SYK", "TBC1D10C", "TBX21", "TCF3", "TFRC", "TGFB1", "THOC1",
    "TICAM1", "TIRAP", "TLR4", "TNFAIP3", "TNFRSF13B", "TNFRSF13C",
    "TNFRSF21", "TNFRSF4", "TNFSF13", "TNFSF13B", "TNFSF4", "TNIP2",
    "TP53BP1", "TPD52", "TRBC1", "TRBC2", "TRDC", "TXLNA", "UNG",
    "URS000075DE64_9606", "VAV3", "VCAM1", "WNT3A", "XBP1", "ZAP70",
    "ZBTB1", "ZFP36L1", "ZFP36L2")

BCR_Act_list_all <- Limma_adj_4[rownames(Limma_adj_4) %in% BCR_act_list,
    ]


# heatmap.3(BCR_Act_list_all, main='BCR_Act_list_all',
# ColSideColors=Col_Ann_Group_All, margins = c(5,9),keysize
# =.6,col=rev(col_spectrum), scale= 'row')
```

```
# legend(x = 'left', c('IgGVP', 'IgAVP', 'IgAVN'),
# col=c('green', 'blue', 'black'), lty=1, lwd = 5, cex=0.8,
# inset = 0, border='black')
```

8.3 What about Mucosal homing?

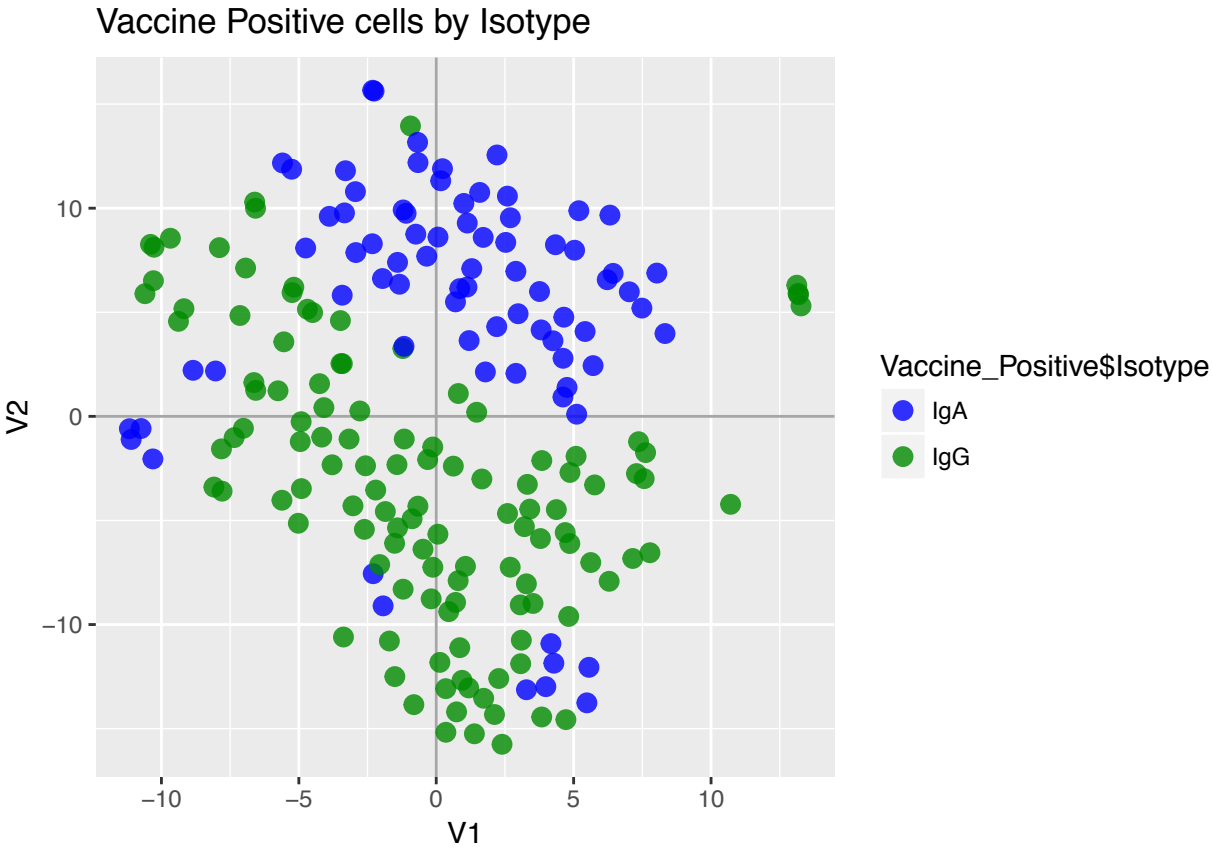
```
MucosalGeneIds <- c("RORA", "CCR9", "CCR10", "CXCR4", "ITGA4", "ITGB7",
                    "IGTB1", "CCR3", "CXCR5", "CCR7", "CCR6", "CCR4", "CLA")
Mucosal_list_DF <- Limma_adj_4[rownames(Limma_adj_4)%in%MucosalGeneIds,]

#heatmap.3(Mucosal_list_DF, main="Mucosal_list_DF",
#ColSideColors=Col_Ann_Group_All,
#margins = c(5,9),keysize =.6,col=rev(col_spectrum),
#scale= 'row')

#legend(x = 'left', c("IgGVP", "IgAVP", "IgAVN"),
#col=c("green", "blue", "black"),
#lty=1, lwd = 5, cex=0.8, inset = 0, border="black")
```

9 Now we subset on QIV positive cells only (IgAVP and IgGVP) and tSNE by Isotype

```
Vaccine_Positive_DF <- Limma_adj_4[,colnames(Limma_adj_4)
                                   %in%rownames(Vaccine_Positive)]
dim(Vaccine_Positive_DF)
```

```
## [1] 11895   195
```

```
set.seed(27)
trans_Vaccine_positive <- as.data.frame(t(Vaccine_Positive_DF))
tSNE_Vaccine_positive <- Rtsne(data.frame(trans_Vaccine_positive),
                               check_duplicates = FALSE, theta = 0.001,
                               perplexity = 30, initial_dims = 10)
library(ggplot2)
ID <- ggplot(data = as.data.frame(tSNE_Vaccine_positive$Y),
             aes(x=V1, y=V2,label = colnames(Vaccine_Positive_DF))) +
        scale_color_manual(values = c('blue', 'green4')) +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vaccine_Positive$Isotype),
                   alpha = 0.8, size = 3)+
        ggtitle("Vaccine Positive cells by Isotype")
ID
```



9.1 Now Identify Differentially expressed genes between the Isotype groups

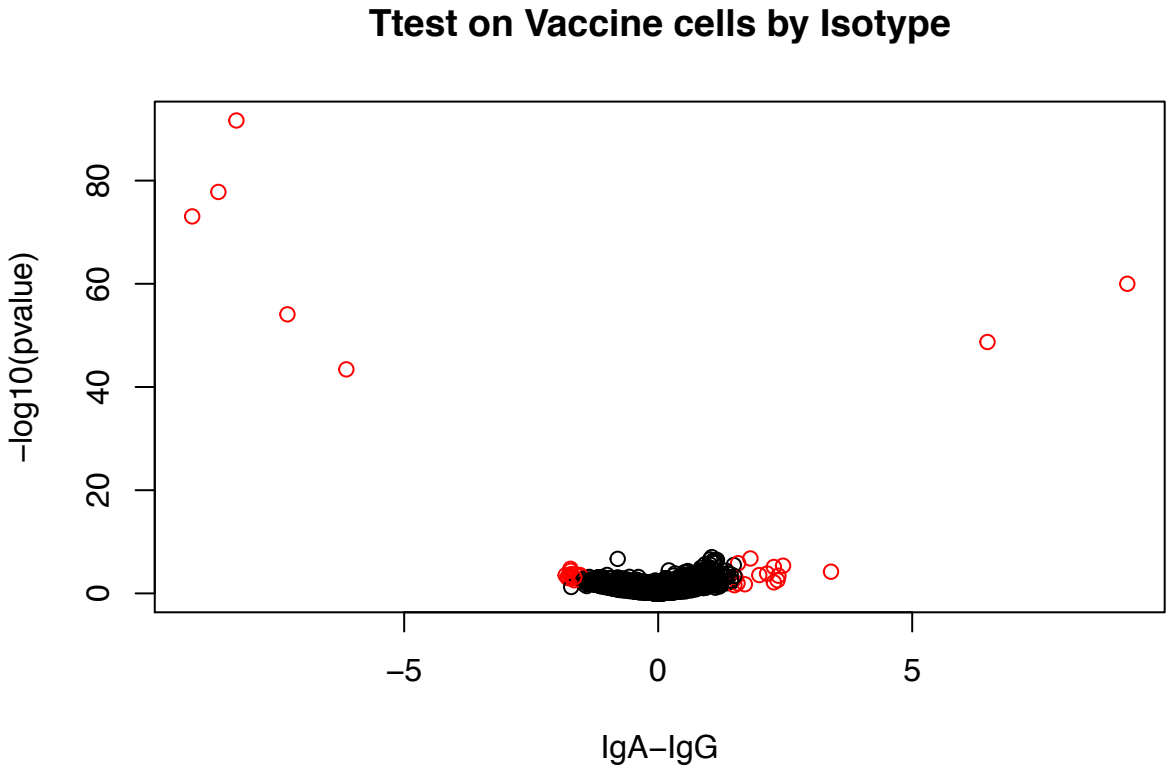
9.2 perform a ttest by Isotype

```
ttest_Vaccine_Positive_DF <- apply(Vaccine_Positive_DF, 1, function(x)
        pairwise.t.test(x, Vaccine_Positive$Isotype,
                        p.adjust.method = 'BH')$p.value)
```

9.3 Make a volcano plot to show fold change vs p value

```
IgA_VP <- subset(Vaccine_Positive, Isotype == 'IgA')
IgA_DF <- Vaccine_Positive_DF[,colnames(Vaccine_Positive_DF)
                              %in% rownames(IgA_VP)]
IgA_mean <- rowMeans(IgA_DF, na.rm=TRUE)

IgG <- subset(Vaccine_Positive, Isotype == 'IgG')
IgG_DF <- Vaccine_Positive_DF[,colnames(Vaccine_Positive_DF)
                              %in% rownames(IgG)]
```

```
IgG_mean <- rowMeans(IgG_DF, na.rm=TRUE)

change_iso <- IgA_mean-IgG_mean
Outcome_iso <- as.data.frame(cbind(IgA_mean, IgG_mean, change_iso,
                                   ttest_Vaccine_Positive_DF))
change_ann_ISO <- abs(Outcome_iso$change_iso) > 1.5
p_ISO <- Outcome_iso$ttest_Vaccine_Positive_DF < .05
ann_ISO <- change_ann_ISO & p_ISO
change_list_ISO <- gsub(TRUE, 'red', ann_ISO)
change_list_ISO <- gsub(FALSE, 'black', change_list_ISO)
plot(Outcome_iso$change_iso, -log10(Outcome_iso$ttest_Vaccine_Positive_DF),
     col=change_list_ISO, ylab='-log10(pvalue)', xlab='IgA-IgG',
     main='Ttest on Vaccine cells by Isotype')
```




```
VP_fc1.5_p.05_Isotype <- Outcome_iso[ann_ISO,]
dim(VP_fc1.5_p.05_Isotype)
```

```
## [1] 37  4
```

```
print(VP_fc1.5_p.05_Isotype)
```

```
##              IgA_mean   IgG_mean change_iso ttest_Vaccine_Positive_DF
## AC233755.1  2.7382354  0.2806458   2.457590              4.317673e-06
## ATP5G1      3.4132800  5.1942278  -1.780948              9.607378e-04
## C1orf56     4.2102411  2.3954409   1.814800              1.696936e-07
## CALM3       3.7972321  5.5748299  -1.777598              6.581658e-04
```

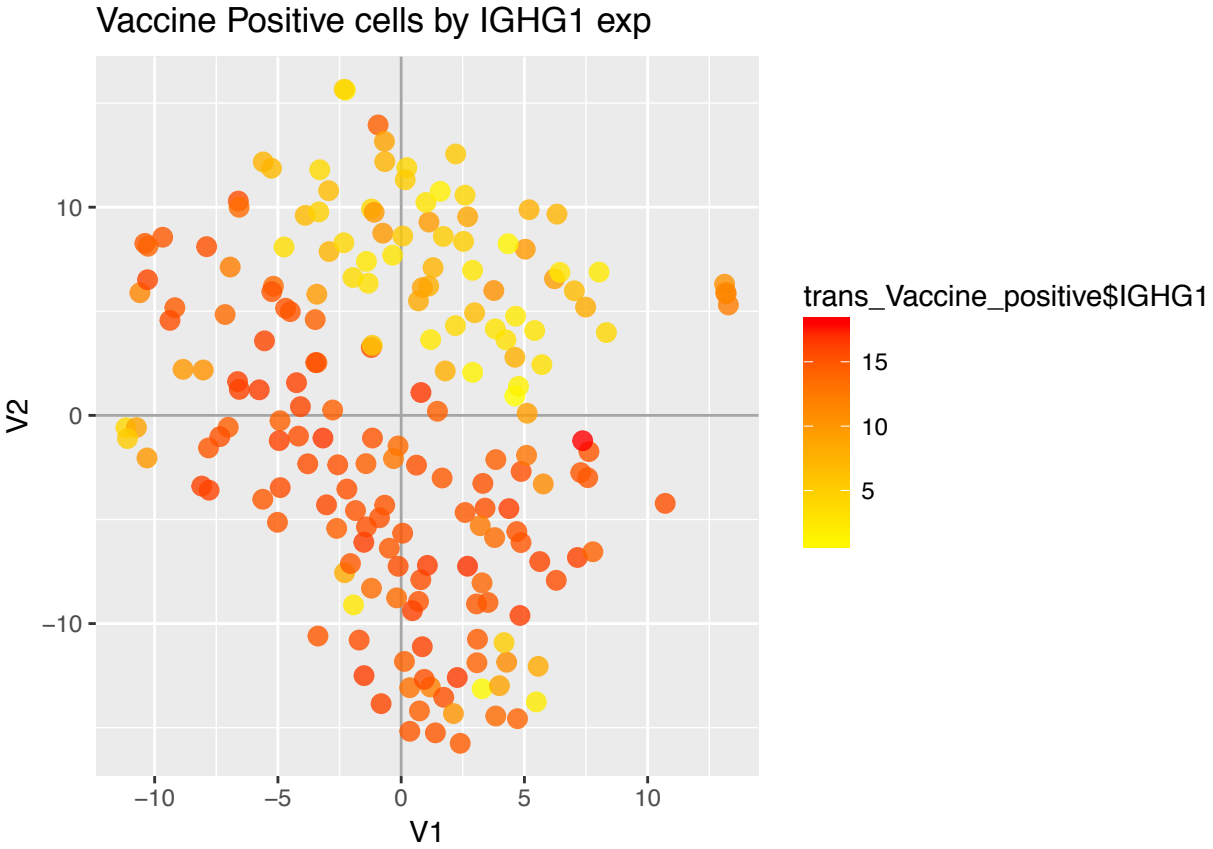
```
## CD53        3.5985293  5.4199901  -1.821461             3.056942e-04
## CHCHD2      5.5334486  7.0419414  -1.508493             3.539422e-03
## EIF2S1      2.2843859  3.9044909  -1.620105             1.265296e-03
## EIF4G2      4.6087300  6.3349946  -1.726265             4.243869e-05
## ENO1        3.2949083  5.0609296  -1.766021             1.095765e-03
## GAPDH       6.6458705  8.3610713  -1.715201             1.851451e-04
## HP1BP3      1.9758036  3.6112904  -1.635487             2.084777e-04
## IFI30       2.3561269  4.0941505  -1.738024             4.383987e-04
## IGHA1      15.0069080  5.7735266   9.233381             9.960145e-61
## IGHA2       8.1001590  1.6186676   6.481491             1.922233e-49
## IGHG1       5.6208523 14.7937234  -9.172871             8.711281e-74
## IGHG2       0.6031112  7.9011154  -7.298004             8.285976e-55
## IGHG3       1.5522590 10.2093244  -8.657065             1.595902e-78
## IGHG4       0.3733102  6.5117327  -6.138422             3.852296e-44
## IGHGP       0.4930180  8.7955943  -8.302576             2.264764e-92
## IGHJ4       6.0466973  3.7694265   2.277271             7.258461e-03
## IGHJ5       3.5751922  1.2089267   2.366266             3.719541e-04
## IGHV3-23    3.9395172  2.3786519   1.560865             1.100931e-02
## IGHV3-43    3.6367066  2.1163599   1.520347             7.792056e-04
## IGHV3-7     2.6540189  0.3776669   2.276352             7.563628e-06
## IGLC2      11.0008832  7.5994095   3.401474             6.233616e-05
## IGLC3       8.4399985  6.0918359   2.348163             2.817535e-03
## IGLJ2       5.2271483  3.5213220   1.705826             1.704878e-02
## IGLJ3       4.3269006  2.8268709   1.500030             2.561747e-02
## IGLV2-11    3.5868805  1.5973582   1.989522             2.833770e-04
## IGLV2-18    1.9834782  0.4093061   1.574172             1.409877e-06
## LGALS1      5.2701208  3.1260941   2.144027             1.440188e-04
## NAPA        1.5995619  3.1795481  -1.579986             3.640357e-04
## NOP10       4.7553040  6.4552963  -1.699992             8.659641e-04
## PSMB8       3.9459908  5.5970062  -1.651015             2.869875e-03
## PTPMT1      0.9341683  2.4736378  -1.539470             2.737704e-04
## SLC38A10    1.6771630  3.4045168  -1.727354             1.716207e-05
## TMEM208     1.8028133  3.4552079  -1.652395             7.833593e-04
```

```
#which of the significant genes are NOT IG genes
VP_Isotype_notIG <- VP_fc1.5_p.05_Isotype[!rownames(
        VP_fc1.5_p.05_Isotype)%in%Kallisto_IG_gene_list,]
dim(VP_Isotype_notIG)
```

```
## [1] 18  4
```

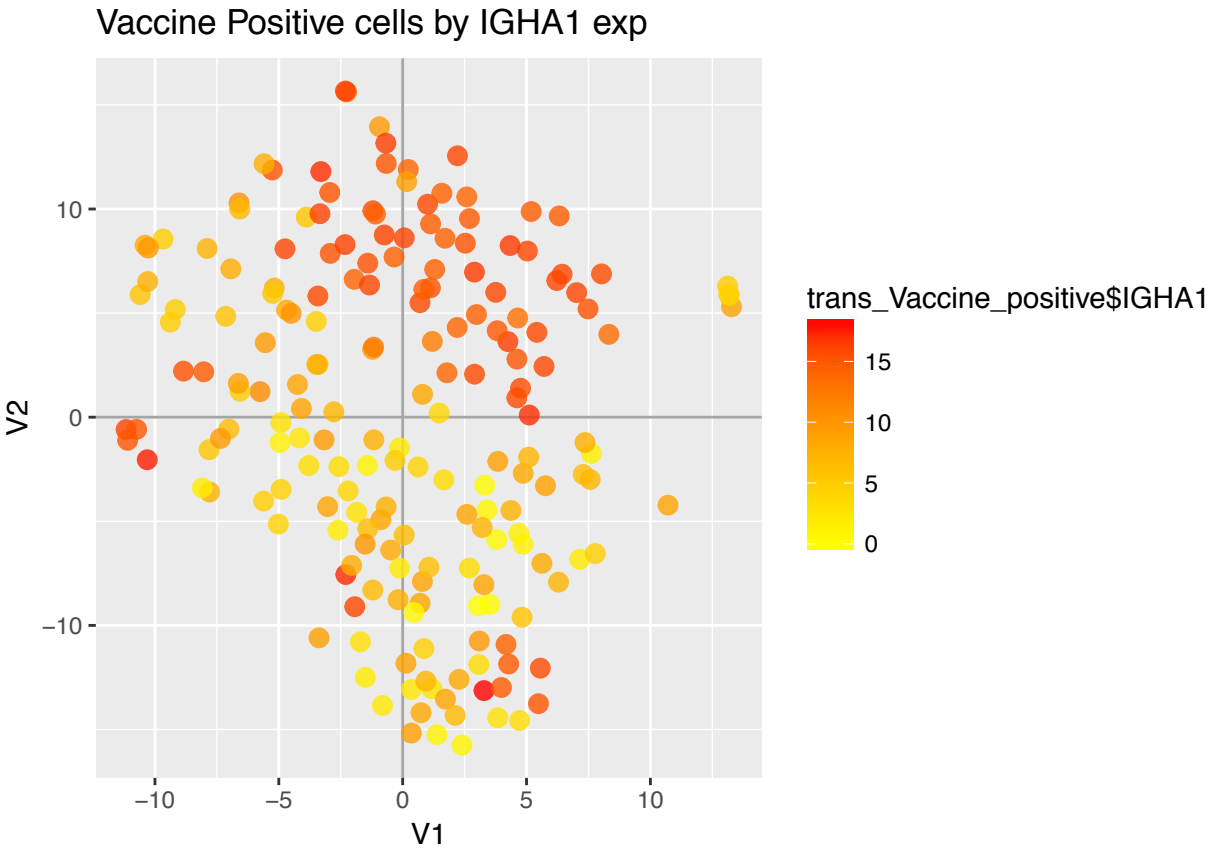
9.4 Show the clusters identified by assigned isotype align with the expression of the IGH constant domains, which are the most significant DEGs identified by the ttest

```
ID <- ggplot(data = as.data.frame(tSNE_Vaccine_positive$Y),
             aes(x=V1, y=V2,label = colnames(Vaccine_Positive_DF))) +
        scale_colour_continuous(low ='yellow', high='red')+
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=trans_Vaccine_positive$IGHG1),
                   alpha = 0.8, size = 3)+
        ggtitle("Vaccine Positive cells by IGHG1 exp")
ID
```




```
ID <- ggplot(data = as.data.frame(tSNE_Vaccine_positive$Y),
             aes(x=V1, y=V2,label = colnames(Vaccine_Positive_DF))) +
        scale_colour_continuous(low ='yellow', high='red')+
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=trans_Vaccine_positive$IGHA1),
                   alpha = 0.8, size = 3)+
        ggtitle("Vaccine Positive cells by IGHA1 exp")
```

ID



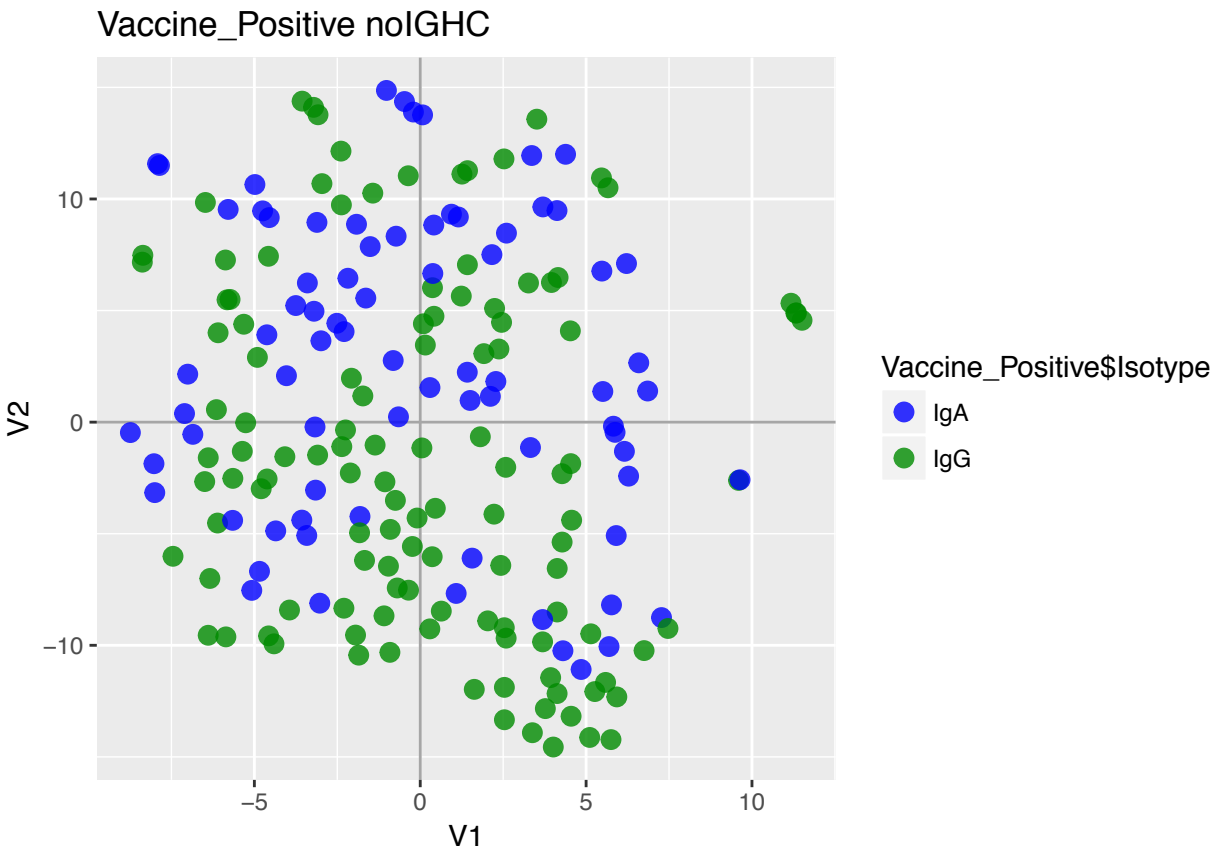
9.5 Now show that clustering is dependent on IGHC genes by removing them

```
IGH_list <- c('IGHG1', 'IGHG2', 'IGHG3', 'IGHG4', 'IGHA1', 'IGHA2',
              'IGHM', 'IGHE', 'IGHD')
Vaccine_Positive_DF_noIGHC <- Vaccine_Positive_DF[!rownames(
        Vaccine_Positive_DF) %in% IGH_list,]
dim(Vaccine_Positive_DF_noIGHC)
```

```
## [1] 11888   195
```

```
trans_Vaccine_positive_noIGHC <- as.data.frame(t(Vaccine_Positive_DF_noIGHC))
tSNE_Vaccine_positive_noIGHC <- Rtsne(data.frame(trans_Vaccine_positive_noIGHC),
                                      check_duplicates = FALSE, theta = 0.001,
                                      perplexity = 30, initial_dims = 10)

ID <- ggplot(data = as.data.frame(tSNE_Vaccine_positive_noIGHC$Y),
             aes(x=V1, y=V2,label = colnames(Vaccine_Positive_DF_noIGHC))) +
        scale_colour_manual(values = c('blue', 'green4'))+
        geom_hline(yintercept = 0, colour = "gray65") +
```

```
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vaccine_Positive$Isotype),
                   alpha = 0.8, size = 3) +
        ggtitle("Vaccine_Positive noIGHC")
ID
```



9.6 Also explore what it looks like when All IG genes are removed

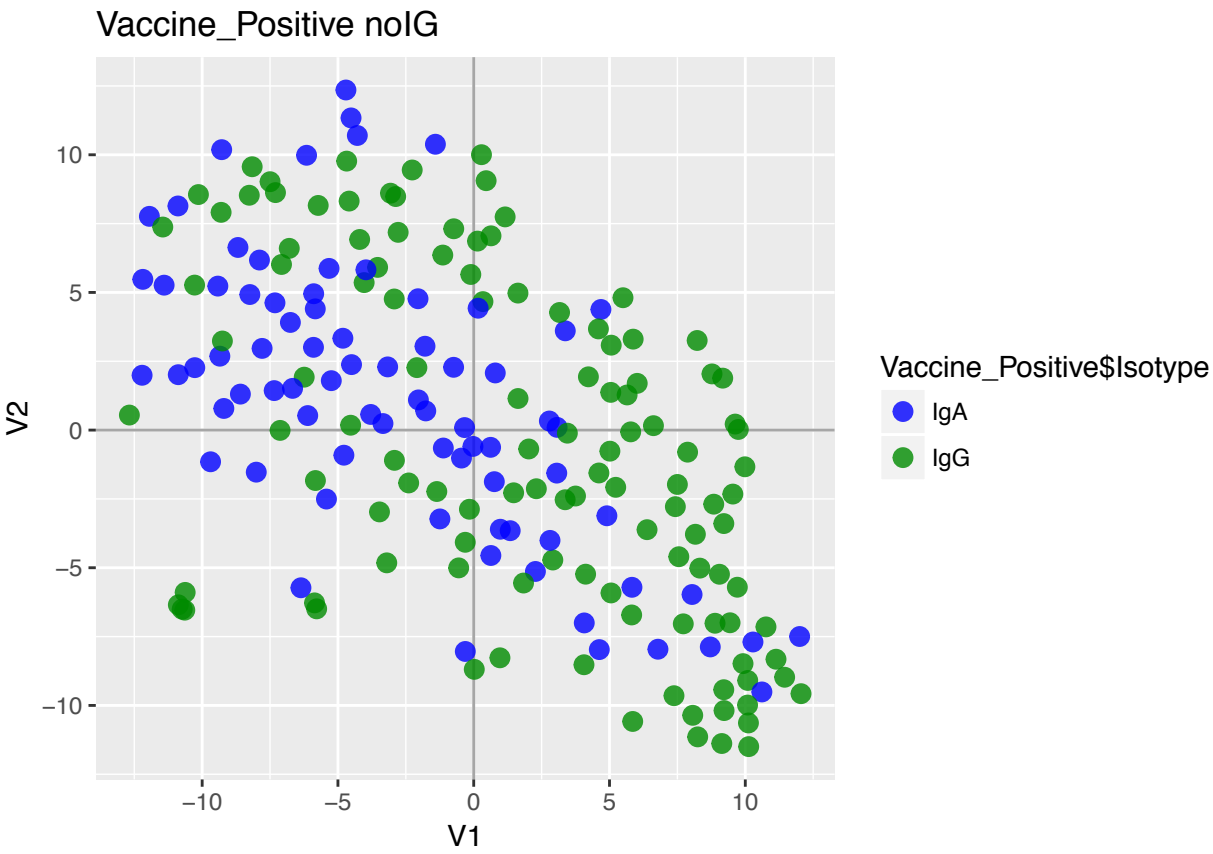
```
Vaccine_Positive_DF_noIG <- Vaccine_Positive_DF[!rownames(
        Vaccine_Positive_DF) %in% Kallisto_IG_gene_list,]
dim(Vaccine_Positive_DF_noIG)
```

```
## [1] 11687   195
```

```
trans_Vaccine_positive_noIG <- as.data.frame(t(Vaccine_Positive_DF_noIG))
tSNE_Vaccine_positive_noIG <- Rtsne(data.frame(trans_Vaccine_positive_noIG),
                                    check_duplicates = FALSE, theta = 0.001,
                                    perplexity = 30, initial_dims = 10)

ID <- ggplot(data = as.data.frame(tSNE_Vaccine_positive_noIG$Y),
             aes(x=V1, y=V2, label = colnames(Vaccine_Positive_DF_noIG))) +
        scale_colour_manual(values = c('blue', 'green4'))+
```

```
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vaccine_Positive$Isotype),
                   alpha = 0.8, size = 3) +
        ggtitle("Vaccine_Positive noIG")
```

ID



9.7 Heatmap of all non-IG DEGs to see what their expression looks like

```
VP_notIG_DEGs <- Vaccine_Positive_DF[rownames(
        Vaccine_Positive_DF)%in%rownames(VP_Isotype_notIG),]
dim(VP_notIG_DEGs)
```

```
## [1]  18 195
```

```
#heatmap.3(VP_notIG_DEGs, main="VP_notIG_DEGs", ColSideColors=ann,
#margins = c(5,9),
#keysize =.6,col=rev(col_spectrum), scale= 'row')
```

```
#legend(x = 'left', c("IgA", "IgG", "clone member", "not in clone"),
#col=c("blue", "green", "red", "grey"), lty=1, lwd = 5,
#cex=0.8, inset = 0, border="black")
```

10 Subset on IgA only and tSNE by Specificity (IgAVP and IgAVN cells)

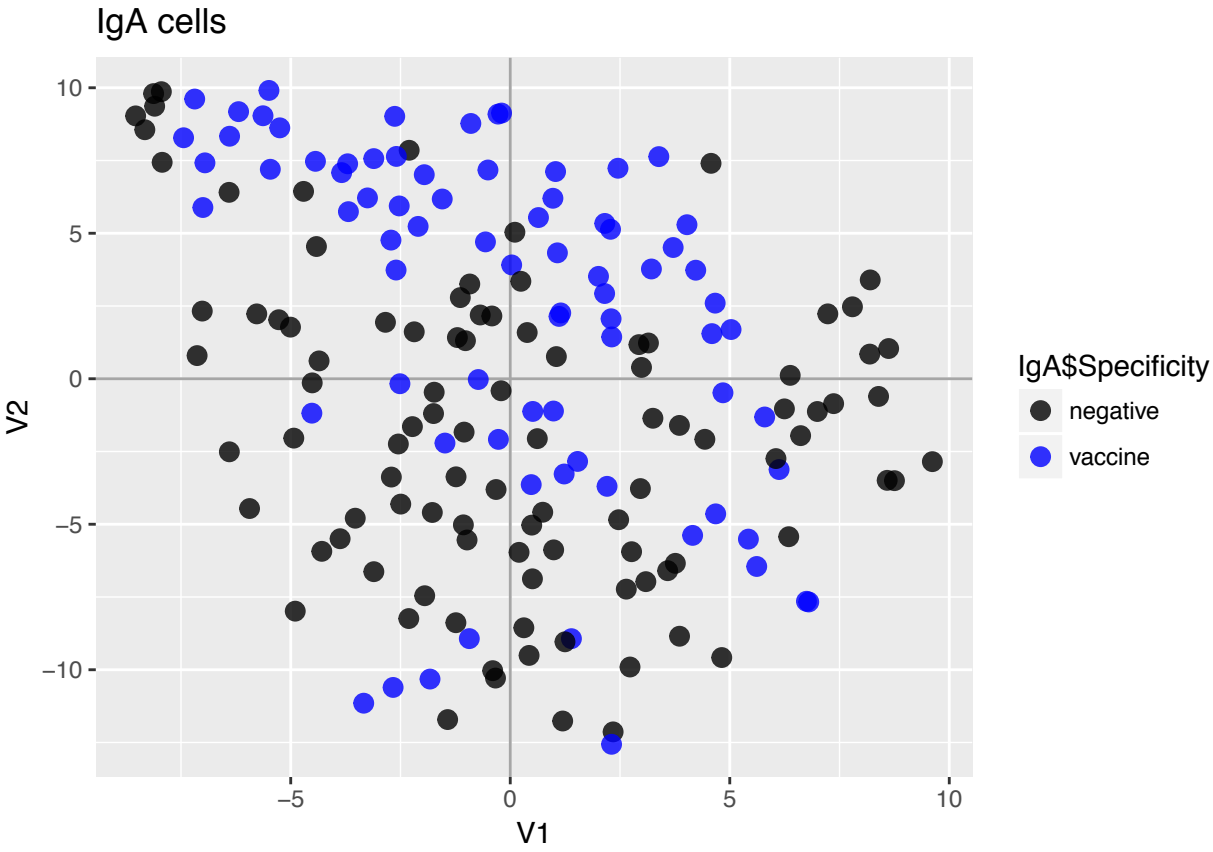
```
IgA <- subset(Vac_annotation_known, Isotype == 'IgA')
dim(IgA)
```

```
## [1] 177   9
```

```
IgA_DF <- Limma_adj_4[,colnames(Limma_adj_4)%in% rownames(IgA)]
dim(IgA_DF)
```

```
## [1] 11895   177
```

```
trans_IgA <- as.data.frame(t(IgA_DF))
tSNE_IgA <- Rtsne(data.frame(trans_IgA),
                  check_duplicates = FALSE, theta = 0.001,
                  perplexity = 30, initial_dims = 10)

ID <- ggplot(data = as.data.frame(tSNE_IgA$Y),
             aes(x=V1, y=V2, label = colnames(IgA_DF)))  +
        scale_colour_manual(values = c('black', 'blue'))+
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=IgA$Specificity),
                   alpha = 0.8, size = 3) +
        ggtitle("IgA cells")
ID
```

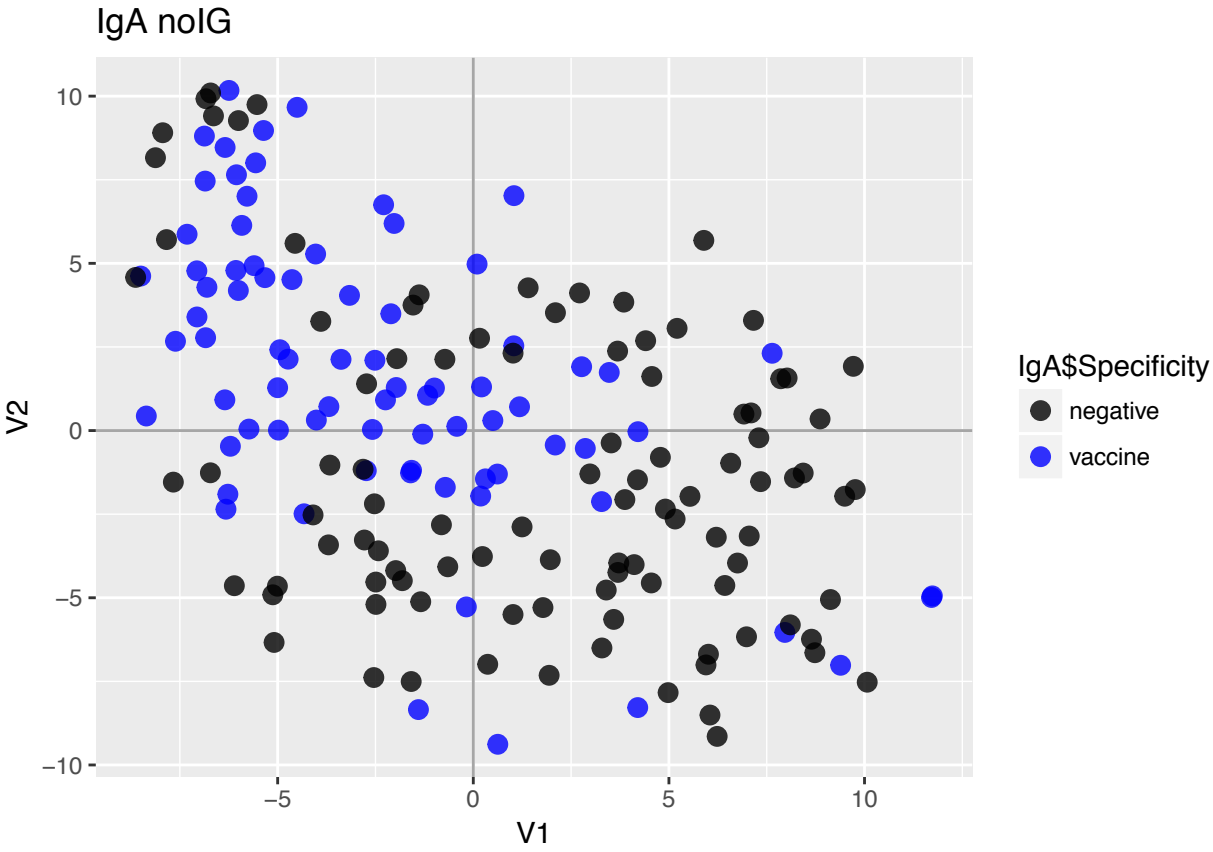


10.1 Repeate tSNE without IG genes

```
IgA_noIG_DF <- limma_adj_noIG[,colnames(limma_adj_noIG)%in% rownames(IgA)]
dim(IgA_noIG_DF) #11687 177
```

```
## [1] 11687   177
```

```
trans_IgA_noIG <- as.data.frame(t(IgA_noIG_DF))
tSNE_IgA_noIG <- Rtsne(data.frame(trans_IgA_noIG), check_duplicates = FALSE,
                       theta = 0.001,
                       perplexity = 30, initial_dims = 10)

ID <- ggplot(data = as.data.frame(tSNE_IgA_noIG$Y),
             aes(x=V1, y=V2, label = colnames(IgA_noIG_DF)))  +
        scale_colour_manual(values = c('black', 'blue'))+
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=IgA$Specificity), alpha = 0.8, size = 3) +
        ggtitle("IgA noIG")
ID
```



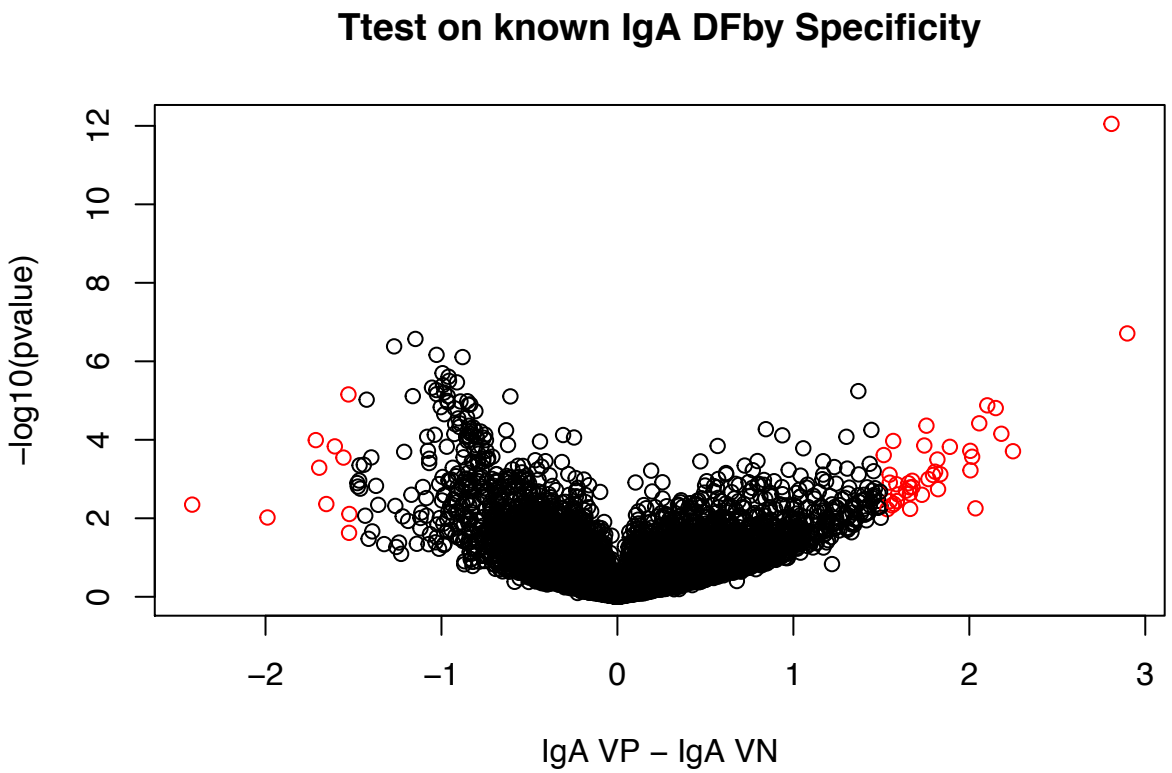
10.2 Perform a ttest by Spec to see how many DEGs are identified

```
ttest_IgA_DF <- apply(IgA_DF, 1, function(x)
        pairwise.t.test(x, IgA$Specificity, p.adjust.method = 'BH')$p.value)
```

10.3 Make volcano plot of Specificity DEGs between IgAVP and IgAVN

```
IgA_VP <- subset(IgA, Specificity == 'vaccine')
IgA_DF_VP<- IgA_DF[,colnames(IgA_DF)%in% rownames(IgA_VP)]
IgA_VP_mean <- rowMeans(IgA_DF_VP, na.rm=TRUE)

IgA_VN <- subset(IgA, Specificity == 'negative')
IgA_DF_VN <- IgA_DF[,colnames(IgA_DF) %in% rownames(IgA_VN)]
IgA_VN_mean <- rowMeans(IgA_DF_VN, na.rm=TRUE)

change_SPEC <- IgA_VP_mean-IgA_VN_mean

Outcome_SPEC <- as.data.frame(cbind(IgA_VP_mean, IgA_VN_mean,
                                    change_SPEC, ttest_IgA_DF))
```

```
change_ann_SPEC <- abs(Outcome_SPEC$change_SPEC) > 1.5
p_SPEC <- Outcome_SPEC$ttest_IgA_DF < .05
ann_SPEC <- change_ann_SPEC & p_SPEC
change_list_SPEC <- gsub(TRUE, 'red', ann_SPEC)
change_list_SPEC <- gsub(FALSE, 'black', change_list_SPEC)
plot(Outcome_SPEC$change_SPEC, -log10(Outcome_SPEC$ttest_IgA_DF),
     col=change_list_SPEC, ylab='-log10(pvalue)', xlab='IgA VP - IgA VN',
     main='Ttest on known IgA DFby Specificity')
```




```
QN_IgA_fc1.5_p.05_SPEC <- Outcome_SPEC[ann_ISO,]
dim(QN_IgA_fc1.5_p.05_SPEC)
```

```
## [1] 37  4
```

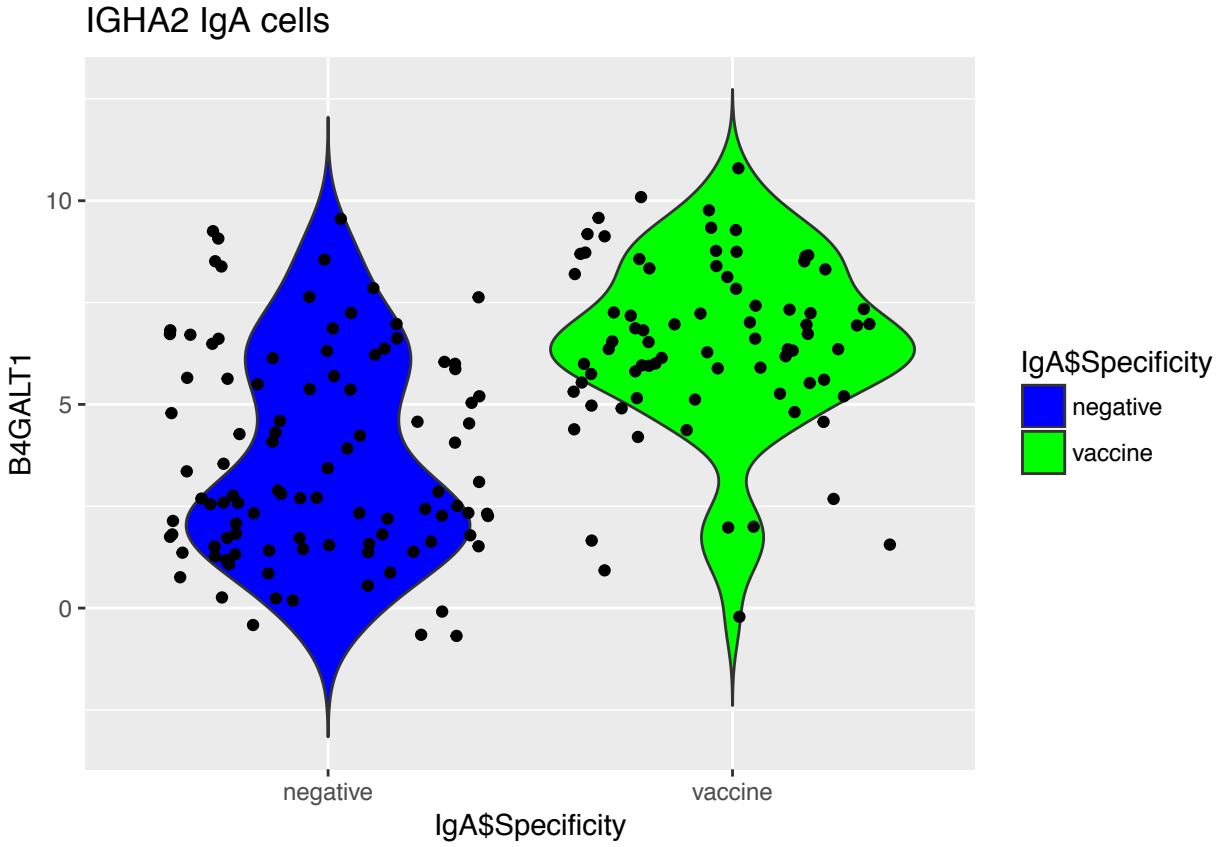
```
#which of the significant genes are NOT IG genes
IgA_SPEC_notIG <- QN_IgA_fc1.5_p.05_SPEC[!rownames(
        QN_IgA_fc1.5_p.05_SPEC)%in%Kallisto_IG_gene_list,]
dim(IgA_SPEC_notIG)
```

```
## [1] 18  4
```

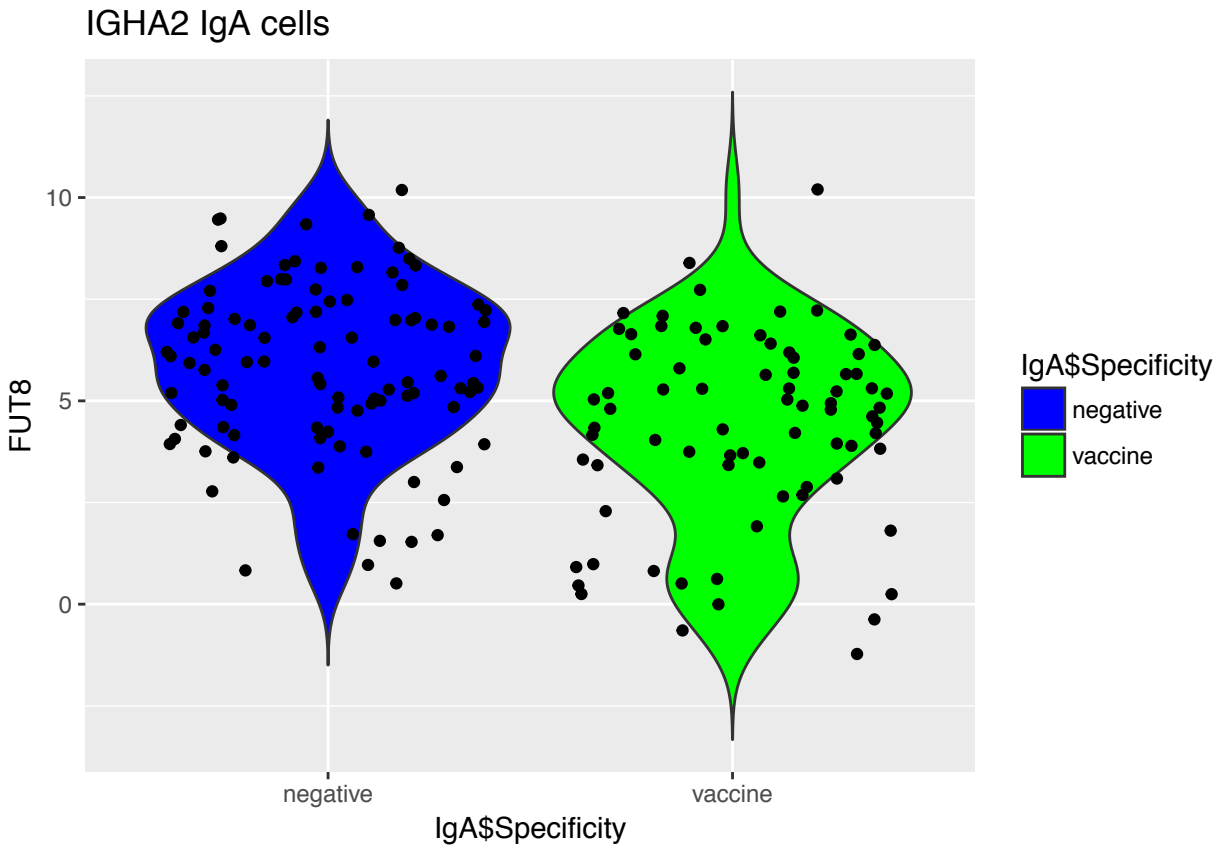
```
IgA_SPEC_IG <- QN_IgA_fc1.5_p.05_SPEC[rownames(
        QN_IgA_fc1.5_p.05_SPEC)%in%Kallisto_IG_gene_list,]
```

10.4 B4GALT1 & FUT8 violin plots

```
t_l <- as.data.frame(t(IgA_DF))
D <- ggplot(t_l, aes(x = IgA$Specificity, y = B4GALT1)) +
        geom_violin(adjust = 1, trim = FALSE, aes(fill=IgA$Specificity)) +
        scale_fill_manual(values=c("blue", "green")) +
        geom_jitter(width=.4, height=.4) +
        ggtitle('IGHA2 IgA cells')
D
```




```
D <- ggplot(t_l, aes(x = IgA$Specificity, y = FUT8)) +
geom_violin(adjust = 1,
            trim = FALSE, aes(fill=IgA$Specificity)) +
        scale_fill_manual(values=c("blue", "green")) +
        geom_jitter(width=.4, height=.4) +
        ggtitle('IGHA2 IgA cells')
D
```



10.5 Repertoire figures without excluding clonal expansions for validation of Specificity DEGs

```
IgA_VP_repertoire <- Vac_repertoire_known[rownames(
        Vac_repertoire_known)%in%colnames(IgA_DF_VP),]
dim(IgA_VP_repertoire)
```

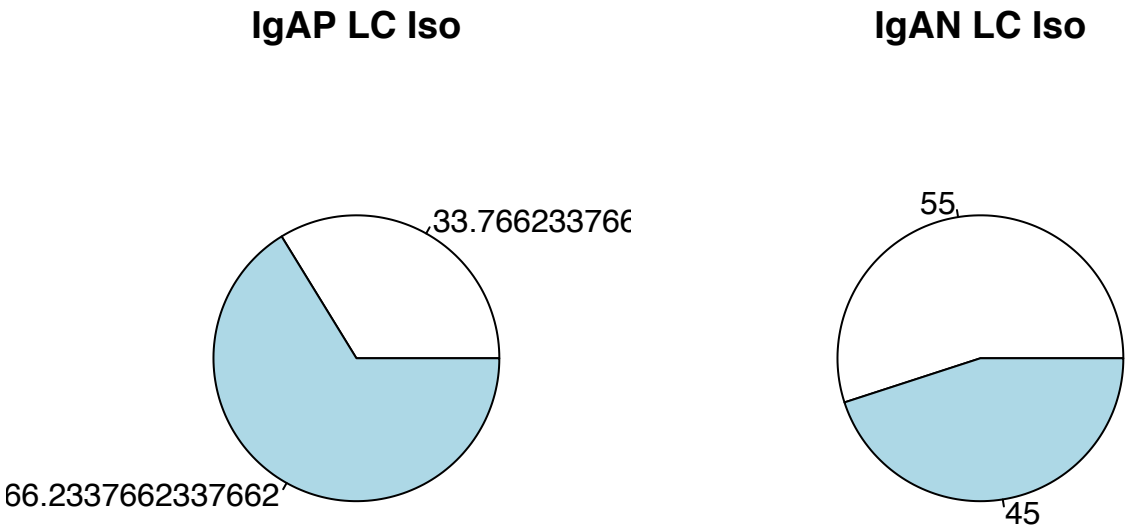
```
## [1] 77 41
```

```
IgA_VN_repertoire <- Vac_repertoire_known[rownames(
        Vac_repertoire_known)%in%colnames(IgA_DF_VN),]
dim(IgA_VN_repertoire)
```

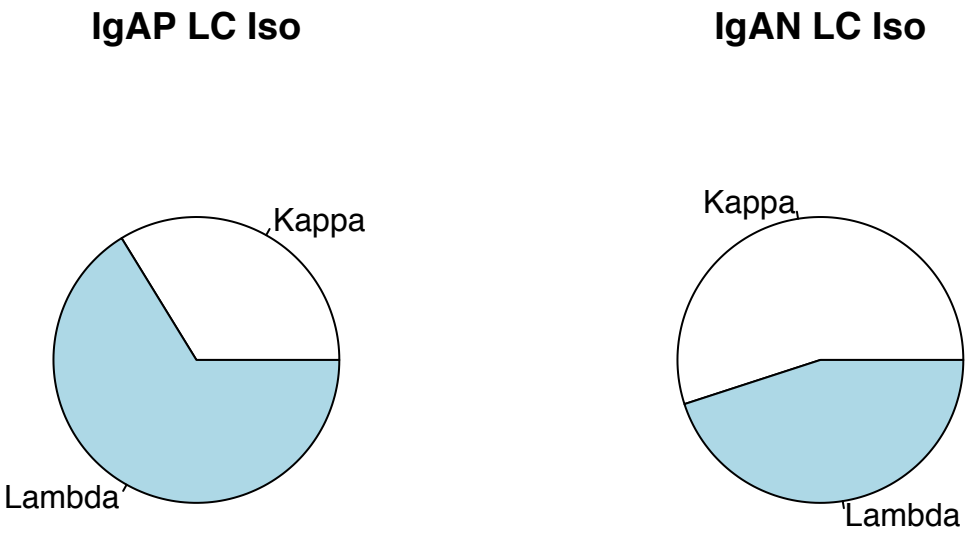
```
## [1] 100  41
```

```
#Light Chain Isotype
LC_Isotype_IgA <- as.data.frame(cbind(IgA_VP = table(
        IgA_VP_repertoire$Isotype),
                                      IgA_VN = table(
                                              IgA_VN_repertoire$Isotype)))
par(mfrow = c(1,2))
```

```
pie(LC_Isotype_IgA$IgA_VP, labels = prop.table(LC_Isotype_IgA$IgA_VP)*100,
    main = 'IgAP LC Iso')
pie(LC_Isotype_IgA$IgA_VN, labels = prop.table(LC_Isotype_IgA$IgA_VN)*100,
    main = 'IgAN LC Iso')
```

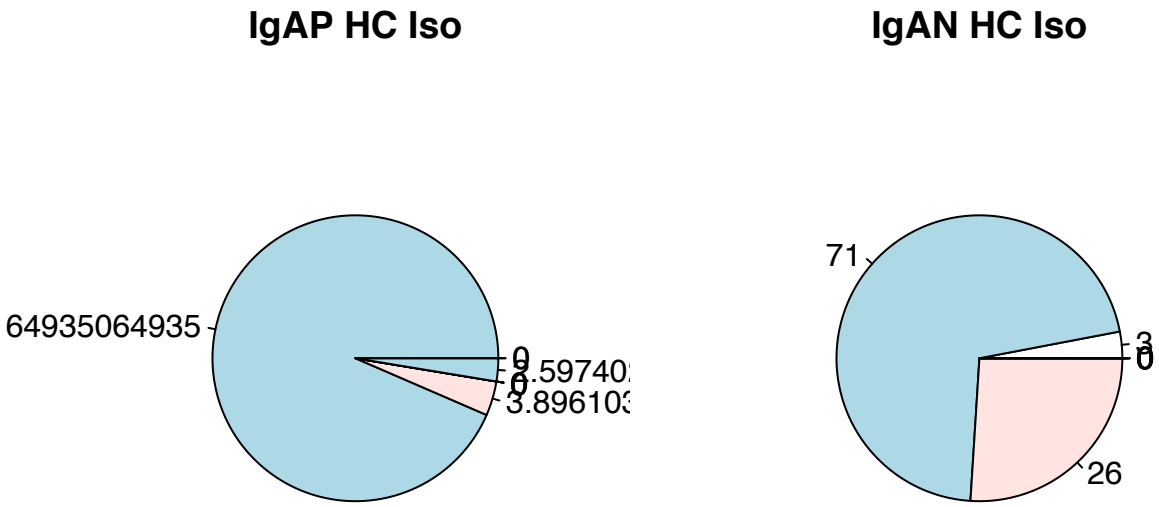



```
pie(LC_Isotype_IgA$IgA_VP, labels = rownames(LC_Isotype_IgA),
    main = 'IgAP LC Iso')
pie(LC_Isotype_IgA$IgA_VN, labels = rownames(LC_Isotype_IgA),
    main = 'IgAN LC Iso')
```

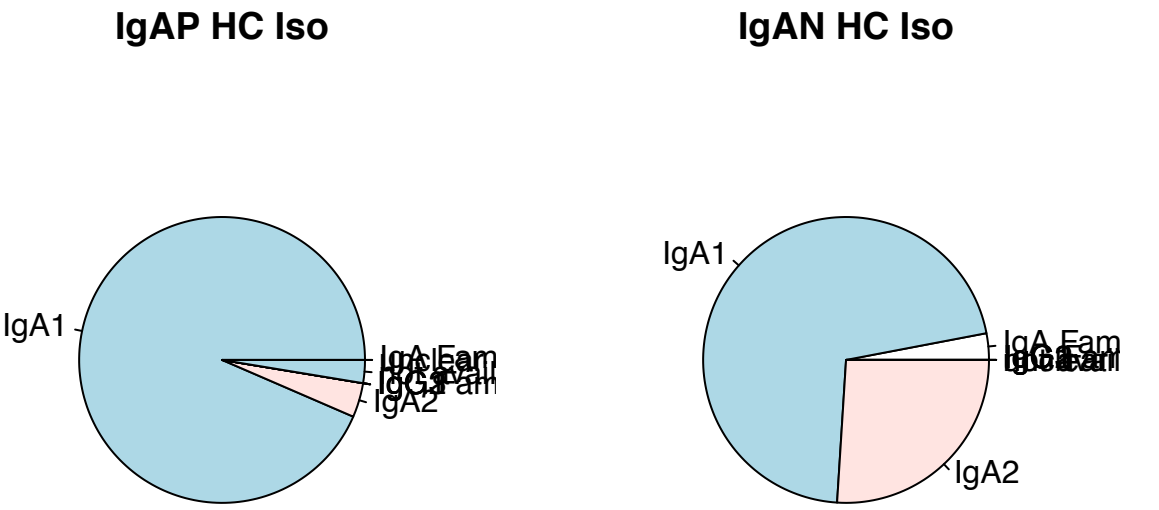



```
#HC Isotype subtype
HC_Isotype_IgA <- as.data.frame(cbind(IgA_VP = table(
        IgA_VP_repertoire$BASIC.Isotype),
                                      IgA_VN = table(
                                              IgA_VN_repertoire$BASIC.Isotype)))
par(mfrow = c(1,2))
```

```
pie(HC_Isotype_IgA$IgA_VP, labels = prop.table(HC_Isotype_IgA$IgA_VP)*100,
    main = 'IgAP HC Iso')
pie(HC_Isotype_IgA$IgA_VN, labels = prop.table(HC_Isotype_IgA$IgA_VN)*100,
    main = 'IgAN HC Iso')
```



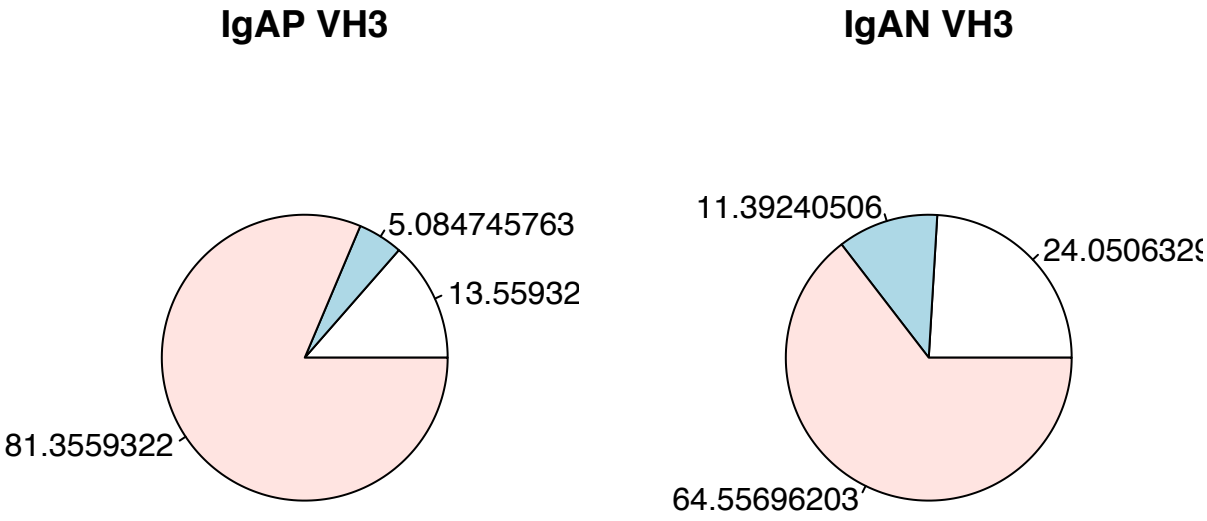

```
pie(HC_Isotype_IgA$IgA_VP, labels = rownames(HC_Isotype_IgA),
    main = 'IgAP HC Iso')
pie(HC_Isotype_IgA$IgA_VN, labels = rownames(HC_Isotype_IgA),
    main = 'IgAN HC Iso')
```



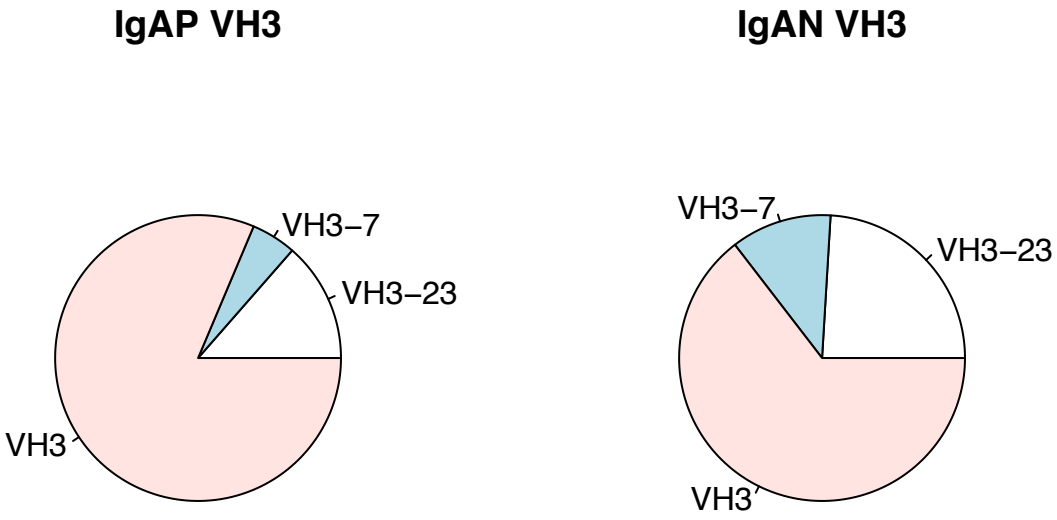
10.6 Look at VH usage, family level summary table made in Excel

```
par(mfrow = c(1,2))

pie(VH3_IgA_usage$IgAP_V, labels = VH3_IgA_usage$IgAP_V,
    main = 'IgAP VH3')
pie(VH3_IgA_usage$IgAN_V, labels = VH3_IgA_usage$IgAN_V,
```

```
    main = 'IgAN VH3')
```




```
pie(VH3_IgA_usage$IgAP_V, labels = rownames(VH3_IgA_usage),
    main = 'IgAP VH3')
pie(VH3_IgA_usage$IgAN_V, labels = rownames(VH3_IgA_usage),
    main = 'IgAN VH3')
```



11 Perform ttest on all three populations by specificity without IG genes And make volcano plot

```
ttest_threepop_SPEC <- apply(limma_adj_noIG, 1, function(x)
       pairwise.t.test(x, Vac_annotation_known$Specificity,
                       p.adjust.method = 'BH')$p.value)

VP <- subset(Vac_annotation_known, Specificity == 'vaccine')
```

```
DF_VP<- limma_adj_noIG[,colnames(limma_adj_noIG)%in% rownames(VP)]
dim(DF_VP)
```

```
## [1] 11687   195
```

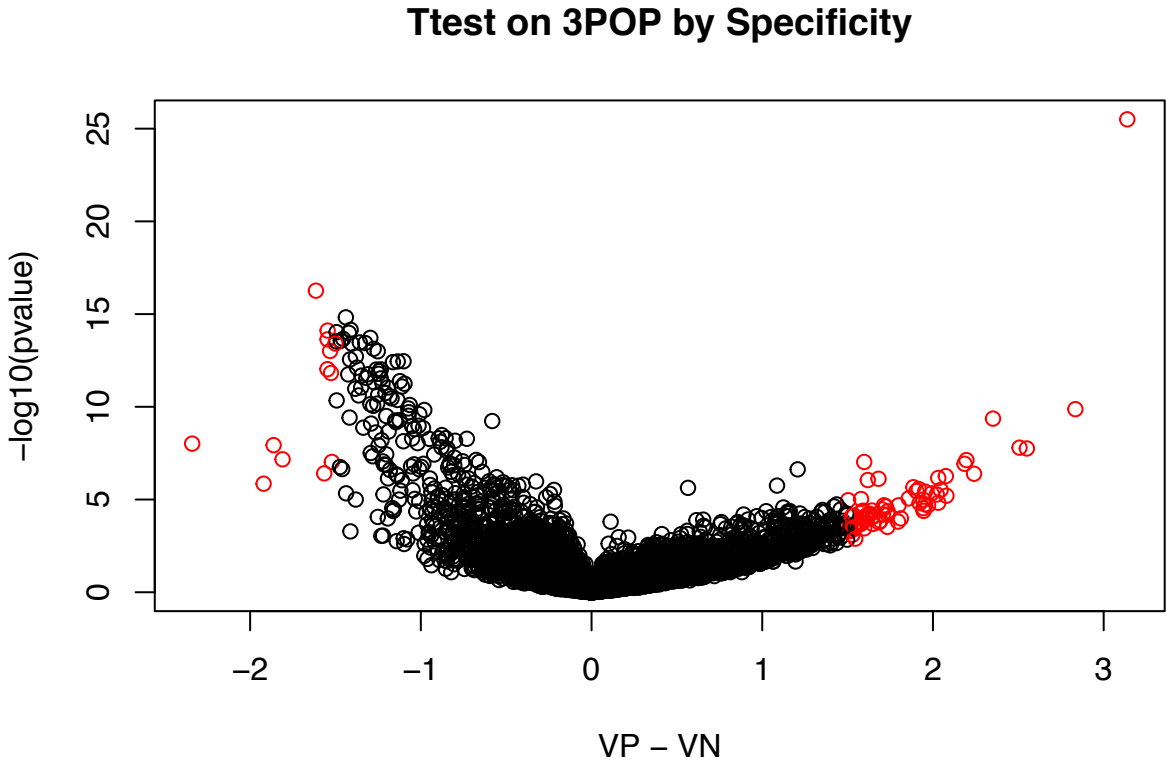
```
VP_mean <- rowMeans(DF_VP, na.rm=TRUE)

VN <- subset(Vac_annotation_known, Specificity == 'negative')
dim(VN)
```

```
## [1] 100   9
```

```
DF_VN <- limma_adj_noIG[,colnames(limma_adj_noIG) %in% rownames(VN)]
dim(DF_VN)
```

```
## [1] 11687   100
```

```
VN_mean <- rowMeans(DF_VN, na.rm=TRUE)

change_3SPEC <- VP_mean-VN_mean

par(mfrow = c(1,1))

Outcome_3SPEC <- as.data.frame(cbind(VP_mean, VN_mean, change_3SPEC,
                                     ttest_threepop_SPEC))
change_ann_3SPEC <- abs(Outcome_3SPEC$change_3SPEC) > 1.5
p_3SPEC <- Outcome_3SPEC$ttest_threepop_SPEC < .05
ann_3SPEC <- change_ann_3SPEC & p_3SPEC
change_list_3SPEC <- gsub(TRUE, 'red', ann_3SPEC)
change_list_3SPEC <- gsub(FALSE, 'black', change_list_3SPEC)
plot(Outcome_3SPEC$change_3SPEC, -log10(Outcome_3SPEC$ttest_threepop_SPEC),
     col=change_list_3SPEC, ylab='-log10(pvalue)', xlab='VP - VN',
     main='Ttest on 3POP by Specificity')
```




```
SPEC_3POP_DEGs <- Outcome_3SPEC[ann_3SPEC,]
dim(SPEC_3POP_DEGs) #89
```

```
## [1] 89  4
```

11.0.1 Still found FUT8 and B4GALT1

12 Clyclone for cell cycle analysis of all three populations

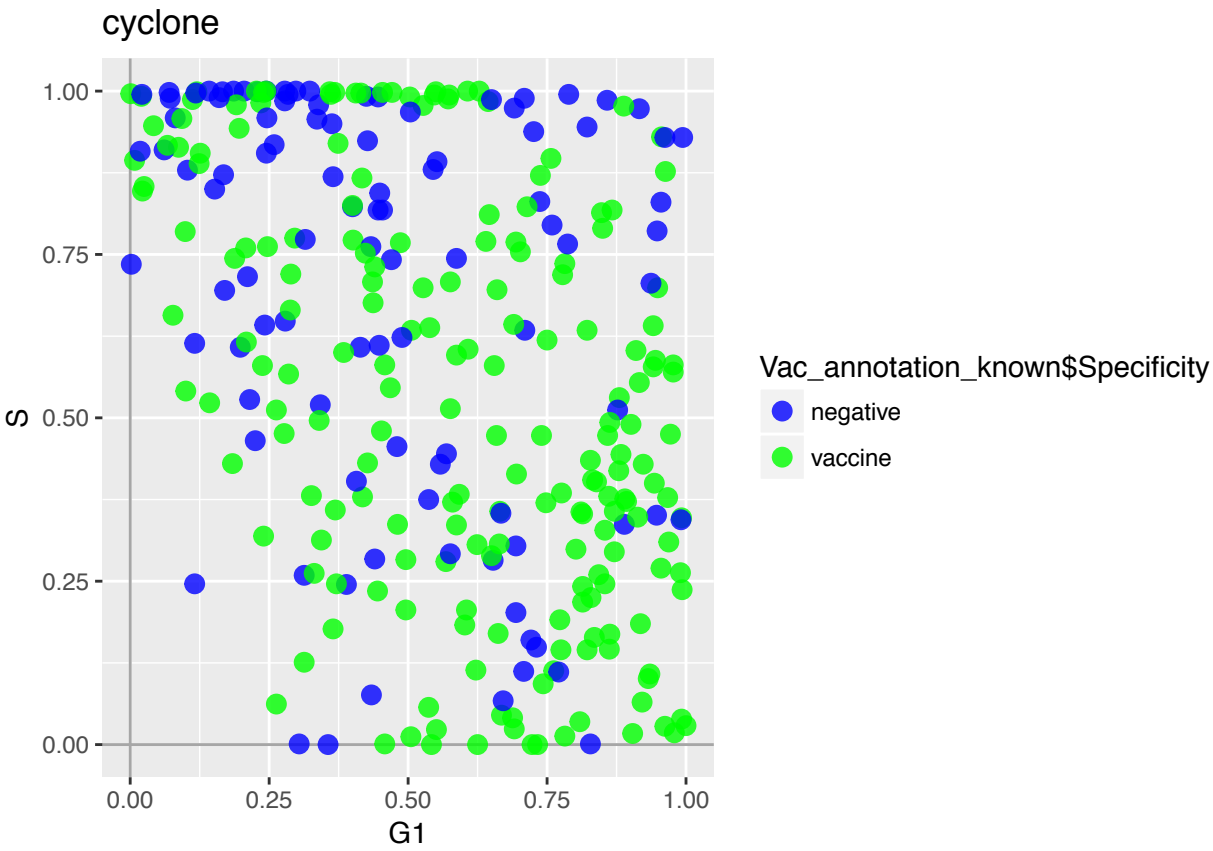
```
hs.pairs <- readRDS(system.file("exdata", "human_cycle_markers.rds",
                                package="scran"))
anno <- select(org.Hs.eg.db, keys=rownames(Limma_adj_4),
               keytype="SYMBOL", column="ENSEMBL")
```

```
## 'select()' returned 1:many mapping between keys and columns
```

```
ensembl <- anno$ENSEMBL[match(rownames(Limma_adj_4), anno$SYMBOL)]

assignments <- cyclone(as.matrix(Limma_adj_4),hs.pairs, gene.names=ensembl)
rownames(assignments$scores) <- colnames(Limma_adj_4)
```

12.1 Make a plot of G1 by S assignment by Specificity

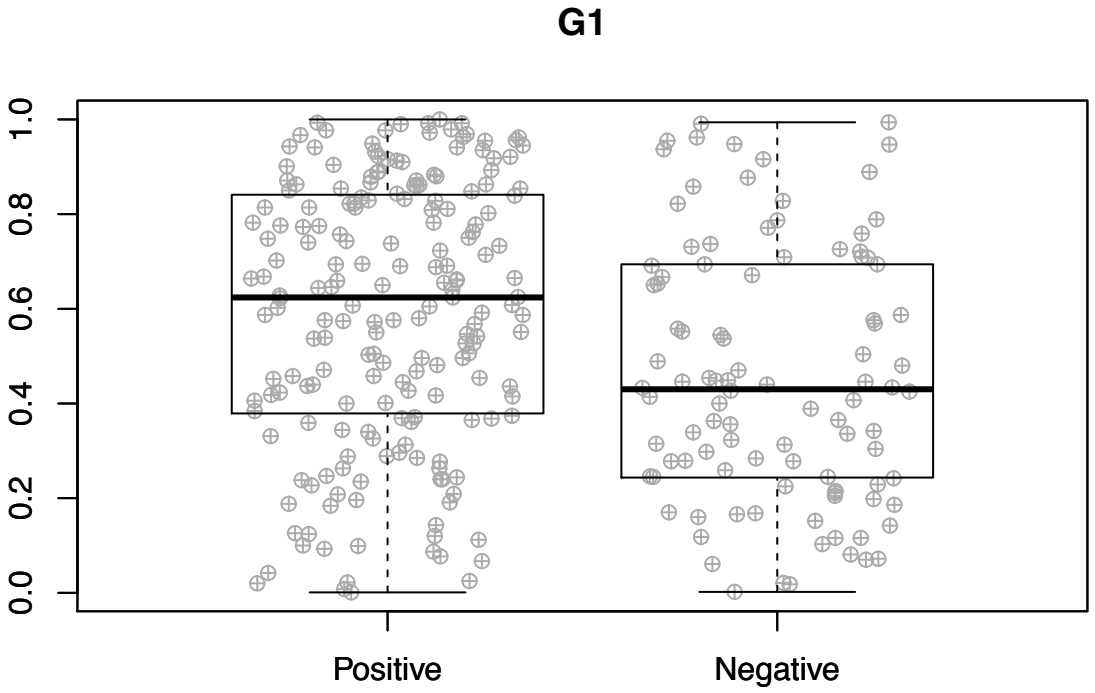
```
ID <- ggplot(data = as.data.frame(assignments$score),
             aes(x=G1, y=S, label = colnames(Limma_adj_4)))   +
        scale_color_manual(values = c('blue', 'green', 'black', 'grey',
                                      'purple')) +
        geom_hline(yintercept = 0, colour = "gray65") +
        geom_vline(xintercept = 0, colour = "gray65") +
        geom_point(aes(colour=Vac_annotation_known$Specificity),
                   alpha = 0.8, size = 3) +
        ggtitle("cyclone")
ID
```



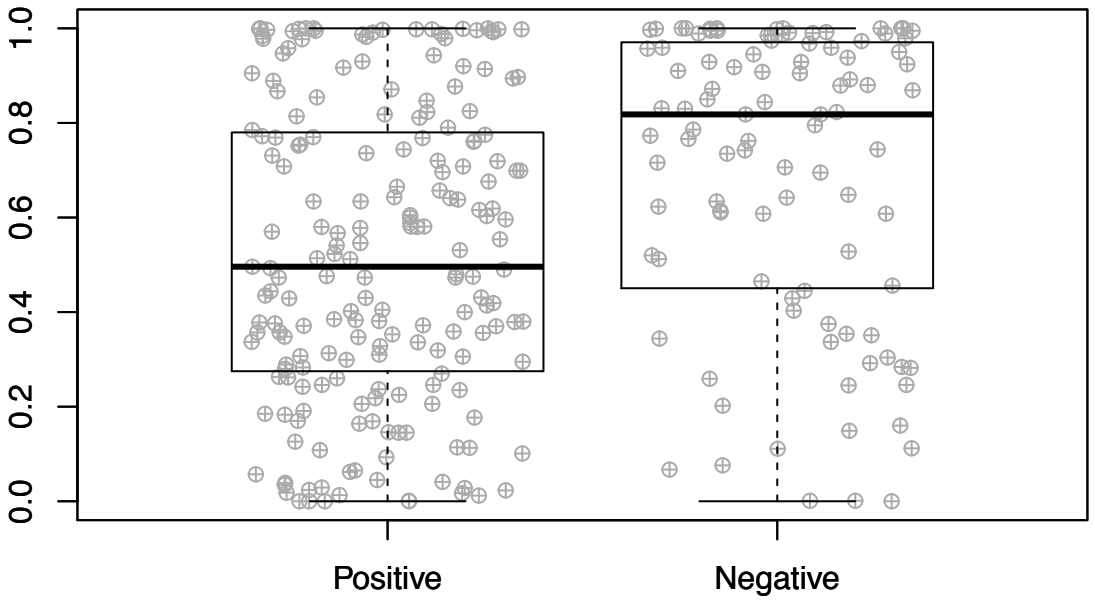
12.2 Then separate Vaccine positie and vaccine negative and make boxplots to compare phase assignment between groups, and perform ttest to assess significance

```
Positive <- subset(Vac_annotation_known, Specificity == 'vaccine')
Negative <- subset(Vac_annotation_known, Specificity == 'negative')

positive_assignments <- assignments$score[rownames(assignments$score)
```

```
                                                %in%rownames(Positive),]
negative_assignments <- assignments$score[rownames(assignments$score)
                                                %in%rownames(Negative),]

Pos_Neg_cycloneG1 <- list('Positive' = positive_assignments$G1,
                          'Negative' = negative_assignments$G1)

stripchart(Pos_Neg_cycloneG1, vertical = TRUE, method = "jitter",
           jitter=.35, pch = 10, col = 'grey67')
boxplot(Pos_Neg_cycloneG1, col='transparent', border = 'black',
        alpha=.8, add = TRUE, main = "G1")
```




```
Pos_Neg_cycloneS <- list('Positive' = positive_assignments$S,'Negative' =
                                  negative_assignments$S)

stripchart(Pos_Neg_cycloneS, vertical = TRUE, method = "jitter",
           jitter=.35, pch = 10, col = 'grey67')
boxplot(Pos_Neg_cycloneS, col='transparent', border = 'black',
        alpha=.8, add = TRUE, main = "S")
```




```
t.test(positive_assignments$G1, negative_assignments$G1)
```

```
##
##  Welch Two Sample t-test
##
## data:  positive_assignments$G1 and negative_assignments$G1
## t = 3.9832, df = 204.1, p-value = 9.455e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.06755818 0.19999362
## sample estimates:
## mean of x mean of y
## 0.5926359 0.4588600
```

```
t.test(positive_assignments$G2M, negative_assignments$G2M)
```

```
##
##  Welch Two Sample t-test
##
## data:  positive_assignments$G2M and negative_assignments$G2M
## t = 0.29727, df = 201.64, p-value = 0.7666
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.05242564  0.07103949
## sample estimates:
## mean of x mean of y
## 0.2060769 0.1967700
```

```
t.test(positive_assignments$S, negative_assignments$S)
```

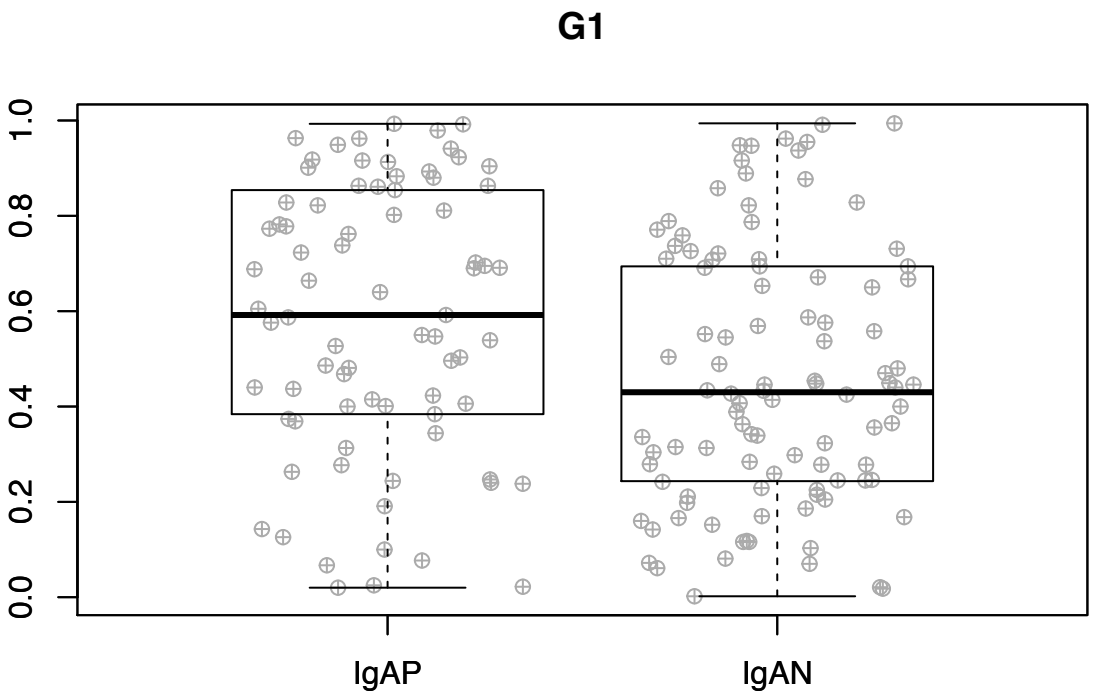
```
##
##  Welch Two Sample t-test
##
## data:  positive_assignments$S and negative_assignments$S
## t = -4.5566, df = 203.33, p-value = 8.96e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.24916187 -0.09865557
## sample estimates:
## mean of x mean of y
## 0.5224513 0.6963600
```

12.3 Repeate cyclone on IgAVN and IgAVP specifically

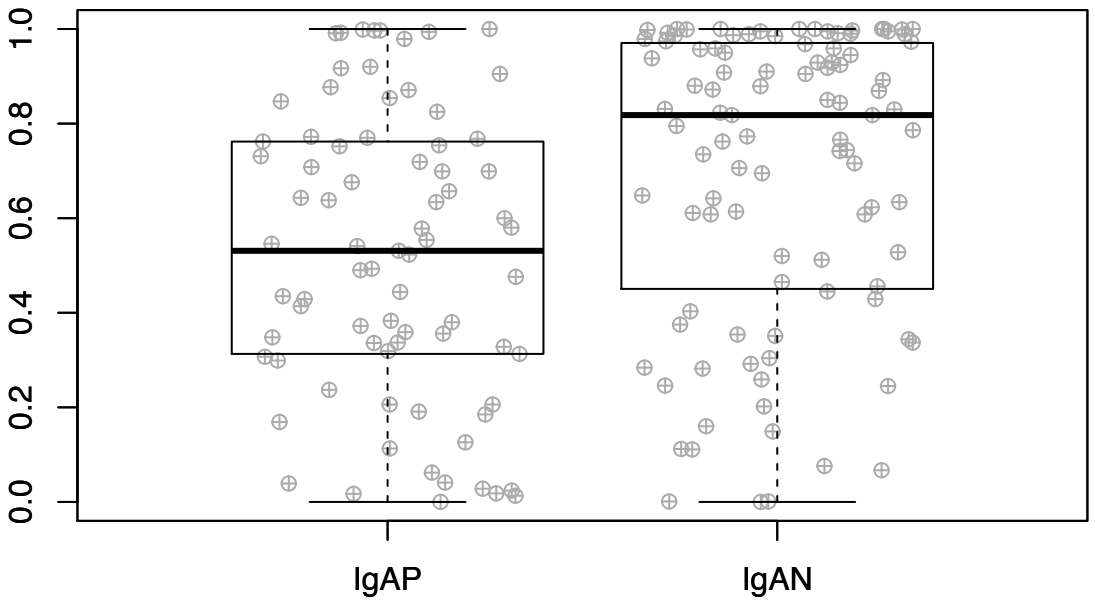
```
Positive_IgA <- subset(Vac_annotation_known, Specificity == 'vaccine'
                       & Isotype == 'IgA')
positive_A_assignments <- assignments$score[rownames(assignments$score)%in%
                                              rownames(Positive_IgA),]

Negative_IgA <- subset(Vac_annotation_known, Specificity == 'negative'
                       & Isotype == 'IgA')
dim(Negative_IgA)
```

```
## [1] 100   9
```

```
negative_A_assignments <- assignments$score[rownames(assignments$score)%in%
                                              rownames(Negative_IgA),]

AP_AN_G1 <- list('IgAP' = positive_A_assignments$G1,
                 'IgAN' = negative_A_assignments$G1)
stripchart(AP_AN_G1, vertical = TRUE, method = "jitter", jitter=.35, pch = 10,
           col = 'grey67')
boxplot(AP_AN_G1, col='transparent', border = 'black', alpha=.8, add = TRUE,
        main = "G1")
```




```
AP_AN_S <- list('IgAP' = positive_A_assignments$S,
                'IgAN' = negative_A_assignments$S)
stripchart(AP_AN_S, vertical = TRUE, method = "jitter", jitter=.35, pch = 10,
           col = 'grey67')
boxplot(AP_AN_S, col='transparent', border = 'black', alpha=.8, add = TRUE,
        main = "S")
```




```
t.test(positive_A_assignments$G1, negative_A_assignments$G1)
```

```
##
##  Welch Two Sample t-test
##
## data:  positive_A_assignments$G1 and negative_A_assignments$G1
## t = 2.9329, df = 159.27, p-value = 0.003854
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.04051112 0.20756108
## sample estimates:
## mean of x mean of y
## 0.5828961 0.4588600
```

```
t.test(positive_A_assignments$G2M, negative_A_assignments$G2M)
```

```
##
##  Welch Two Sample t-test
##
## data:  positive_A_assignments$G2M and negative_A_assignments$G2M
## t = 0.59217, df = 155.64, p-value = 0.5546
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.05610961  0.10415402
## sample estimates:
## mean of x mean of y
## 0.2207922 0.1967700
```

```
t.test(positive_A_assignments$S, negative_A_assignments$S)
```

```
##
##  Welch Two Sample t-test
##
## data:  positive_A_assignments$S and negative_A_assignments$S
## t = -3.7839, df = 164.52, p-value = 0.0002159
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2673391 -0.0840043
## sample estimates:
## mean of x mean of y
## 0.5206883 0.6963600
```