ANALYSIS OF THE INHOUSE HNSC DATASET

#R version 3.6.1 (2019-07-05) – "Action of the Toes" #Copyright (C) 2019 The R Foundation for Statistical Computing #Platform: x86_64-w64-mingw32/x64 (64-bit)

PART 0: PREPARE THE ENVIRONMENT ——————————————

#1. Load (or install) all necessary packages.

```r
library("readxl")
library("writexl")
library("readr")
library("ComplexHeatmap")
library("circlize")
library("ggplot2")
library("gridExtra")
library("gplots")
library("mclust")
library("GSEABase")
library("methods")
library("edgeR")
library("geneplotter")
library("genefilter")
library("BiocGenerics")
library("Biobase")
library("graph")
library("XML")
library("lattice")
library("limma")
library("shinythemes")
library("shiny")
library("RColorBrewer")
library("parallel")
library("cluster")
library("Matrix")
library("locfit")
library("snow")
library("GSVA")
library("dplyr")
library("ggplot2")
library("data.table")
library ("remotes")
library("plyr")
library("magrittr")
library("OIsurv")
library("survival")
library("KMsurv")
library("splines")
library("survminer")
library("ggpubr")
library("survutils")
library("scales")
library("ggpubr")
library("tidyverse")
library("corrr")
library("igraph")
library("ggraph")
library("tidygraph")
library("CoxBoost")
library("glmnet")
library("randomForest")
library("class")
library("dml")
library("MASS")
library("readr")
library("Rtsne")
library("stats")
library("ggridges")
library("gdata")
library("ggrepel")
library("corrplot")
library("ggExtra")
library("gridExtra")
library("rstatix")
library("ggpubr")
library("viridis")
library("corrplot")
library("xlsx")
library("plotly")
library("plot3D")
library("tidyr")
```

PART 1: IMPORT THE FILES FOR THE ANALYSIS ————————————————————

#1. Import the TGFB and ALTEJ genelists.

```r
TGFBgeneset <- read.table("Input/TGFB_list.txt", quote="\"", comment.char="", stringsAsFactors=FALSE)
ALTEJgeneset <- read.table("Input/ALTEJ_list.txt", quote="\"", comment.char="", stringsAsFactors=FALSE)
```

#2. Turn the genelistS into lists and create a list containing both of them.

```r
TGFBlist <- as.list(TGFBgeneset[,1:1]) #55 genes (50 + synonyms)
ALTEJlist <- as.list(ALTEJgeneset[,1:1]) #45 genes (36 + synonyms)
Bothgenelists <- list(TGFBUPgeneset=TGFBlist, ALTEJgeneset=ALTEJlist)
```

#3. Import the file with gene expression information.

```
Exp <- read_excel("Input/HNSC_explants_allinfo.xlsx", sheet = "Expression_(log2_&_groupedZsc_t")
##This file contatins normalized gene expression of the HNSC explants.
##Normalization (as described in Methods): Normalized in the nSolver (by Qi Liu) -> Log2 transformed ->
```

#4. Import the file with the phenotype information.

```
DF <- read_excel("Input/HNSC_explants_allinfo.xlsx",  na = "N/A")
##This file contains the    Characteristics of the HNSC explants. Such as:
  ##their HPV status, tissue origin, % of pSmad2+ cells,
  ##% of 53BP1+ cells (cells with >5 foci at 5h after 5Gy irradiation or after Olaparib),
  ##mean expression of the TGFB genes, mean expression of the alt-EJ genes, and BAlt score.
```

PART 2: DATA WRANGLING

#1.Data description.

```
dim(Exp) #47 samples.
dim(DF) #47 samples.
table(DF$treatments) #6 treated with a TGFB inhibitor (LY).
Duplicated <- DF[duplicated(DF$Sample_ID), ] #10 duplicated samples. Excluding the ones treated with LY
```

#2. Exclusion of samples treated with a TGFB inhibitor, given that those are not part of the present study.

```
LY <- DF[which(DF$treatments=="LY2157299"),]
LY <- as.list(LY$Nanostring_ID)
DF <- DF[-which(DF$treatments=="LY2157299"),] #47->41s.
Exp <- Exp[,-which(colnames(Exp) %in% LY)] #47->41s.
```

#3. Exclusion of the duplicated samples, given that the replicates are not part of the present study.

```
Duplicated <- as.list(Duplicated$Nanostring_ID)
DF <- DF[-which(DF$Nanostring_ID %in% Duplicated),] #41->37s.
Exp <- Exp[,-which(colnames(Exp) %in% Duplicated)] #41->37s.
```

#4. Tidy the HPV status variable.

```
DF$HPV_Status <- ifelse(is.na(DF$HPV_Status), "NA", DF$HPV_Status)
```

#5. Explore the dataset.

```
table(DF$HPV_Status) #17-, 7+, 13 NA.
```

```
##
## HPV- HPV+   NA
##   17    7   13
```

```r
table(DF$Tissue) #22 from patients, 15 from PDX.
```

```
##
## patient     PDX
##      22      15
```

```r
#Summary: 37 HNSC tumor explants (22 from patients and 15 from PDX).
```

#6. See how many samples have pSMAD2 and 53BP1 information.

```r
summary(DF$`pSMAD2_%`) #2 samples without this information (NA).
summary(DF$`5Gy_% (>5 foci)`) #18 samples without this information (NA).
```

#7. Export the cleaned files.

```r
write.xlsx(Exp, file="Output/Expression_HNSC_explants.xlsx", col.names = TRUE, row.names = TRUE)
write.xlsx(DF, file="Output/Phenotype_HNSC_explants.xlsx", col.names = TRUE, row.names = TRUE)
```

PART 3: CLUSTERING HEATMAP ————————————————

——

#1. Create a matrix with only TGFB and ALTEJ genes expression values.

```r
Miniexp <- Exp[which(Exp$Signature.x == "TGFB" | Exp$Signature.x == "ALTEJ"),] #86 genes.
table(Miniexp$Signature.x) #50 TGFB genes, 36 ALTEJ genes.
rownames(Miniexp) <- Miniexp$...1
Miniexp1 <- as.matrix(Miniexp[,-(1:2)])
rownames(Miniexp1) <- rownames(Miniexp)
Miniexp <- Miniexp1
class(Miniexp)<-"numeric"
```

#2. Create row annotations: Signature.

```r
Signature <- data.frame(Gene=rownames(Miniexp))
Signature$Signature <- ifelse(Signature$Gene %in% TGFBlist, "TGFB", "ALTEJ")
rownames(Signature) <- Signature$Gene
Signature$Gene <- NULL
Signature_Colors=list(Signature=c("TGFB"="hotpink2", "ALTEJ"="seagreen3"))
Signature_Annotation = HeatmapAnnotation(df  = Signature, col = Signature_Colors, which = "row")
```

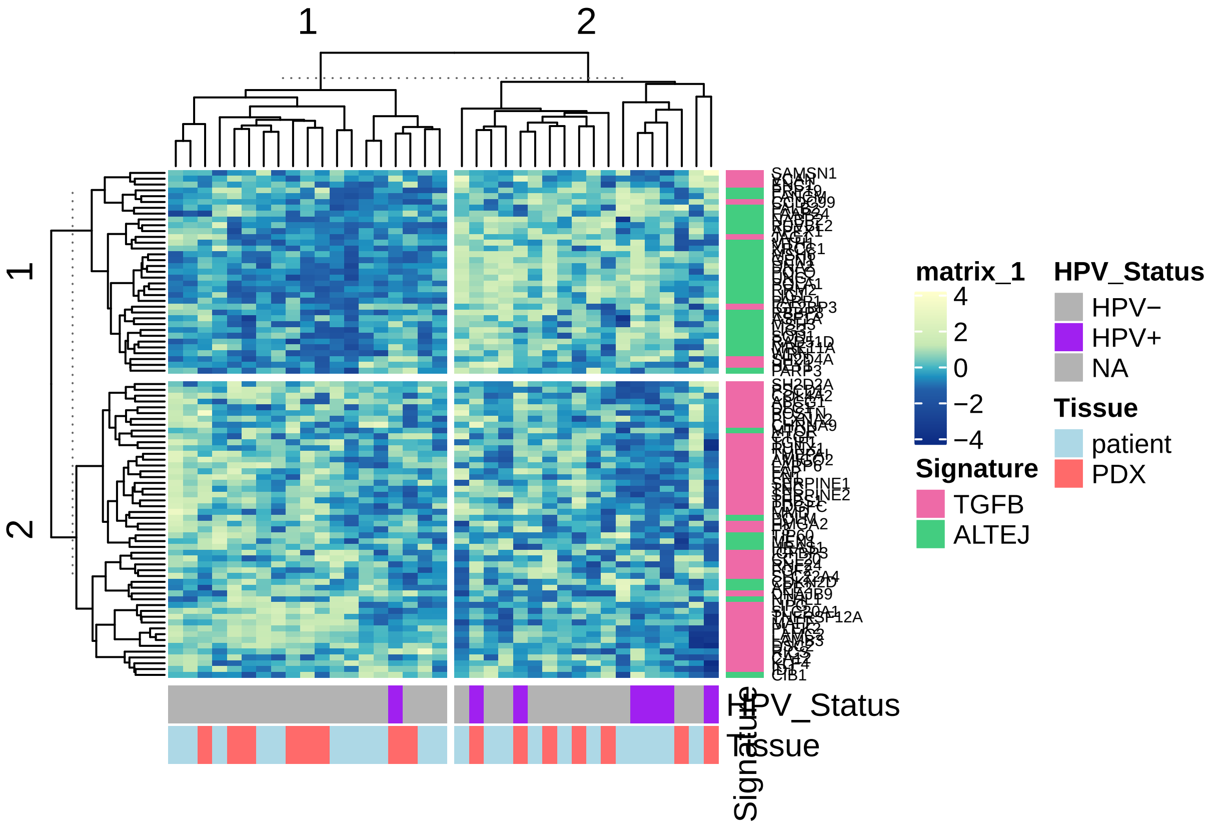
#3. Create column annotations: tissue origin + HPV status.

```r
Sampleinfo <- DF
rownames(Sampleinfo) <- Sampleinfo$Nanostring_ID
Sampleinfo <- Sampleinfo[match(colnames(Miniexp), rownames(Sampleinfo)), ] #Order samples as in the Min
rownames(Sampleinfo) <- Sampleinfo$Nanostring_ID
names(Sampleinfo)
Sampleinfo <- Sampleinfo[,c("HPV_Status", "Tissue", "Nanostring_ID")] #Select the necessary variables.
```

```
rownames(Sampleinfo) <- Sampleinfo$Nanostring_ID
Sampleinfo$Nanostring_ID <- NULL
Sampleinfo <- as.data.frame(Sampleinfo)
summary(Sampleinfo)
Sample_Colors=list(HPV_Status=c("HPV-"="grey70", "HPV+"="purple", "NA"="grey70"),
                   Tissue=c("patient"="light blue", "PDX"="indianred1")) #Choose colors for the annotat
Sample_Annotations = HeatmapAnnotation(df = Sampleinfo, col = Sample_Colors, which = "column")
```

#4. Create a hierarchical clustering heatmap.

```
Heatmap = Heatmap(Miniexp,
                  clustering_distance_rows = "euclidean",
                  clustering_method_rows = "ward.D2",
                  row_dend_width = unit(1.5, "cm"),
                  clustering_distance_columns = "euclidean",
                  clustering_method_columns = "ward.D2",
                  column_dend_height = unit(1.5, "cm"),
                  row_names_gp = gpar(fontsize = 6.2), #Or fontsize=6.
                  show_column_names = FALSE,
                  column_km = 2,
                  row_km = 2,
                  bottom_annotation = Sample_Annotations,
                  right_annotation = Signature_Annotation,
                  colorRamp2(c(4, 1.2, 0.5, 0, -0.5, -1.2, -4), brewer.pal(7,"YlGnBu"))); Heatmap #PDF
```



```
                    #colorRamp2(c(4, 1.1, 0.5, 0, -0.5, -1.1, -4), brewer.pal(7,"RdBu"))); Heatmap
```

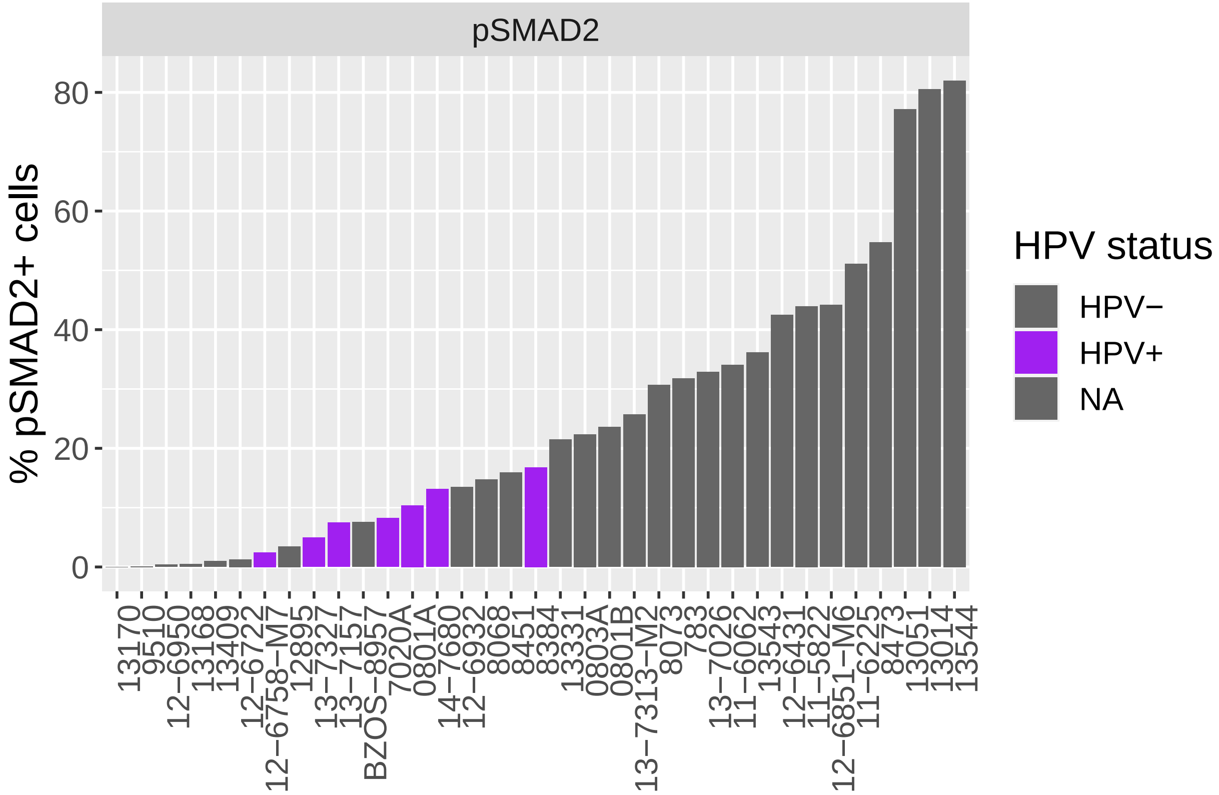
PART 4: BARPLOTS

#1. Order the samples by their pSMAD2 values (that is, by their % of pSmad2 positive cells).

```
DF1 <- DF[order(DF$`pSMAD2_%`), ]
DF1$Sample_ID <- as.factor(DF1$Sample_ID)
DF1$Sample_ID <- factor(DF1$Sample_ID, levels = DF1$Sample_ID[order(DF1$`pSMAD2_%`)])
```

#2. Barplot of pSMAD2.

```
DF1 <- DF1[-which(is.na(DF1$`pSMAD2_%`)),]
DF1$title <- "pSMAD2"
ggplot(DF1, aes(fill=HPV_Status, y=`pSMAD2_%`, x=Sample_ID)) +
  geom_bar(position="stack", stat="identity") + #position = "fill" for percentages, "stack" for counts.
  labs(x = " ", y = "% pSMAD2+ cells", fill = "HPV status") +
  theme(axis.text.x=element_text(angle=90,hjust=1)) +
  scale_fill_manual(values=c("grey40", "purple", "grey40")) +
  theme(text = element_text(size=15)) +
  facet_grid(. ~ title, scales="free_x")
```

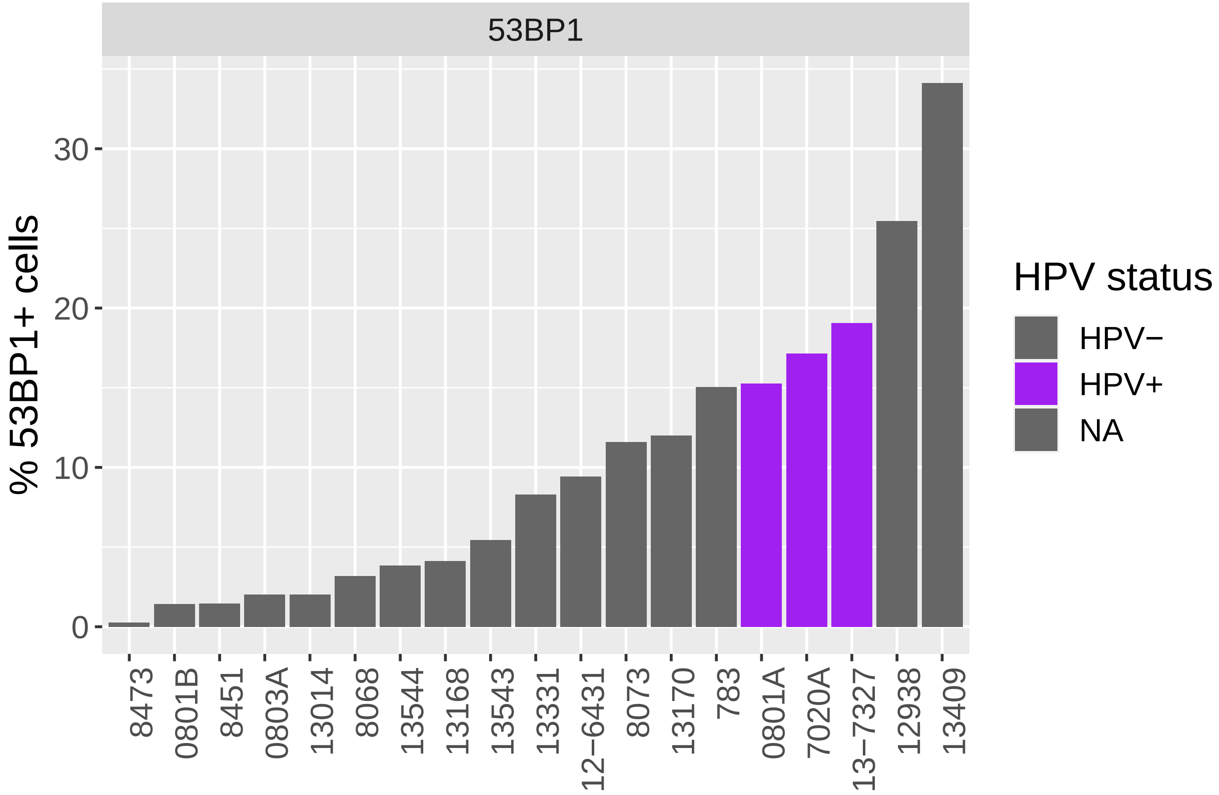


#3. Order samples by 53BP1 values.

```
DF1 <- DF[order(DF$`5Gy_% (>5 foci)`), ]
DF1$Sample_ID <- as.factor(DF1$Sample_ID)
DF1$Sample_ID <- factor(DF1$Sample_ID, levels = DF1$Sample_ID[order(DF1$`5Gy_% (>5 foci)`)])
```

#4. Barplot of 53BP1.

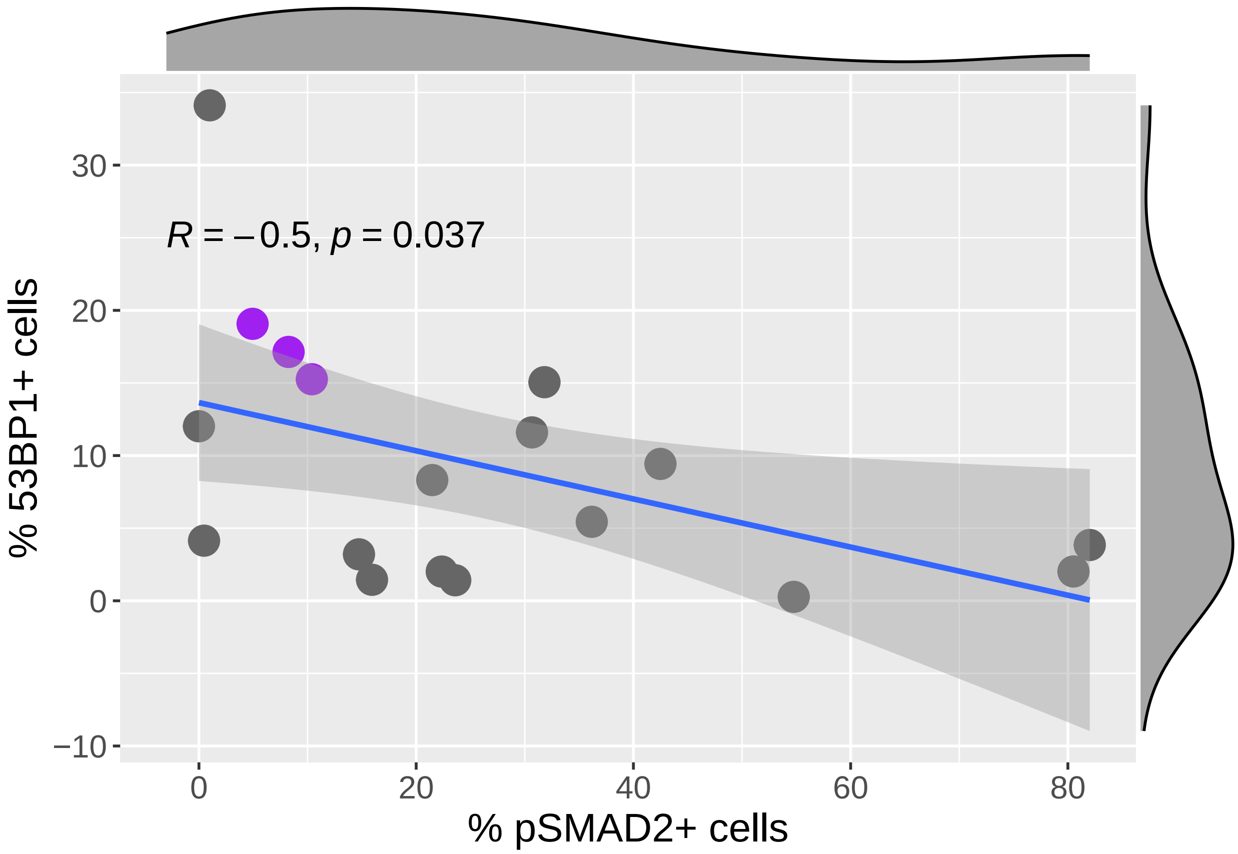
```
DF1 <- DF1[-which(is.na(DF1$`5Gy_% (>5 foci)`)),]
DF1$title <- "53BP1"
ggplot(DF1, aes(fill=HPV_Status, y=`5Gy_% (>5 foci)`, x=Sample_ID)) +
  geom_bar(position="stack", stat="identity") + #position = "fill" for percentages, "stack" for counts.
  labs(x = " ", y = "% 53BP1+ cells", fill = "HPV status") +
  theme(axis.text.x=element_text(angle=90,hjust=1)) +
  scale_fill_manual(values=c("grey40", "purple", "grey40")) +
  theme(text = element_text(size=15)) +
  facet_grid(. ~ title, scales="free_x")
```



PART 5: SCATTERPLOTS

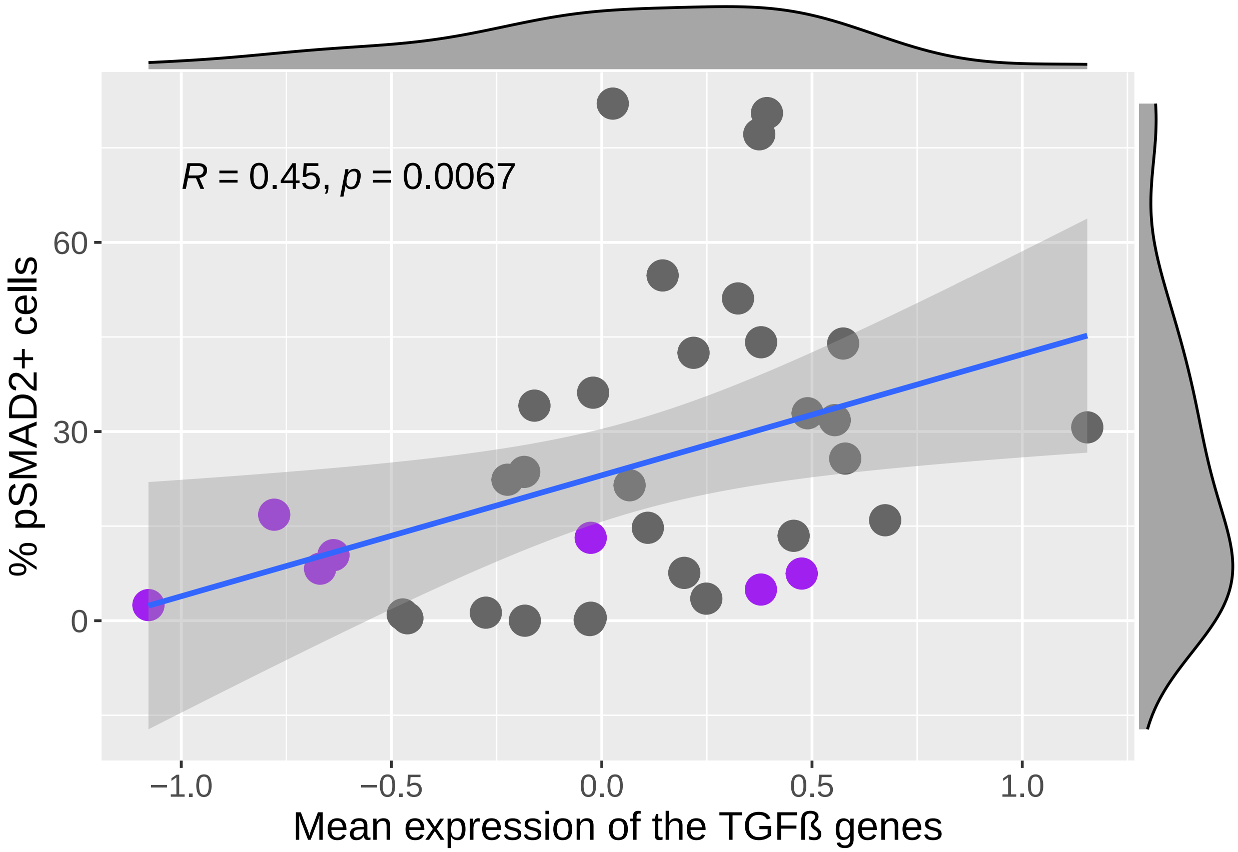
#1. Scatterplot of PSMAD2 versus 53BP1 scatterplot with HPV color.

```
p <- ggplot(DF, aes(x=`pSMAD2_%`, y=`5Gy_% (>5 foci)`)) +
  geom_point(size=5, aes(color=HPV_Status)) +
  scale_color_manual(values=c("grey40", "purple", "grey40")) +
  labs(x = "% pSMAD2+ cells", y = "% 53BP1+ cells", col="HPV status") +
  geom_smooth(method="glm", fullrange=TRUE) +
  stat_cor(label.x=-3, label.y=25, method="spearman", size=5) +
  theme(text = element_text(size=15)) +
  theme(legend.title=element_text(size=8))
p <- p + rremove("legend")
p1 <- ggMarginal(p,  size=10, fill = "grey65"); p1
```



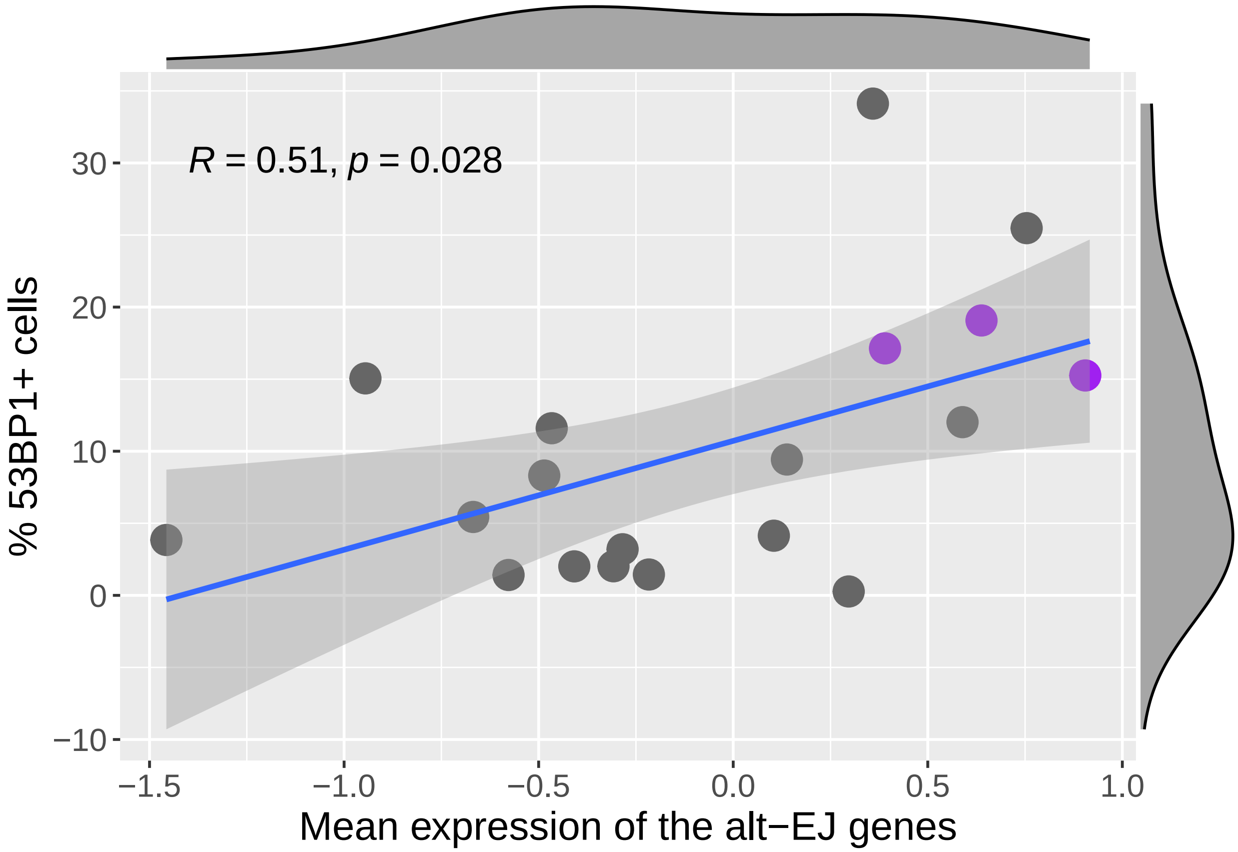
#2. Scatterplot of TGFB vs PSMAD2 with HPV color.

```
p <- ggplot(DF, aes(x=Mean_TGFB, y=`pSMAD2_%`)) +
  geom_point(size=5, aes(color=HPV_Status)) +
  scale_color_manual(values=c("grey40", "purple", "grey40")) +
  labs(x = "Mean expression of the TGF  genes", y = "% pSMAD2+ cells", col="HPV status") +
  geom_smooth(method="glm", fullrange=TRUE) + #Add correlation line
  stat_cor(label.x=-1, label.y=70, method="spearman", size=5) +
  theme(text = element_text(size=15)) +
  theme(legend.title=element_text(size=8))
p <- p + rremove("legend")
p1 <- ggMarginal(p,  size=10, fill = "grey65"); p1 #PDF 5x6.5.
```



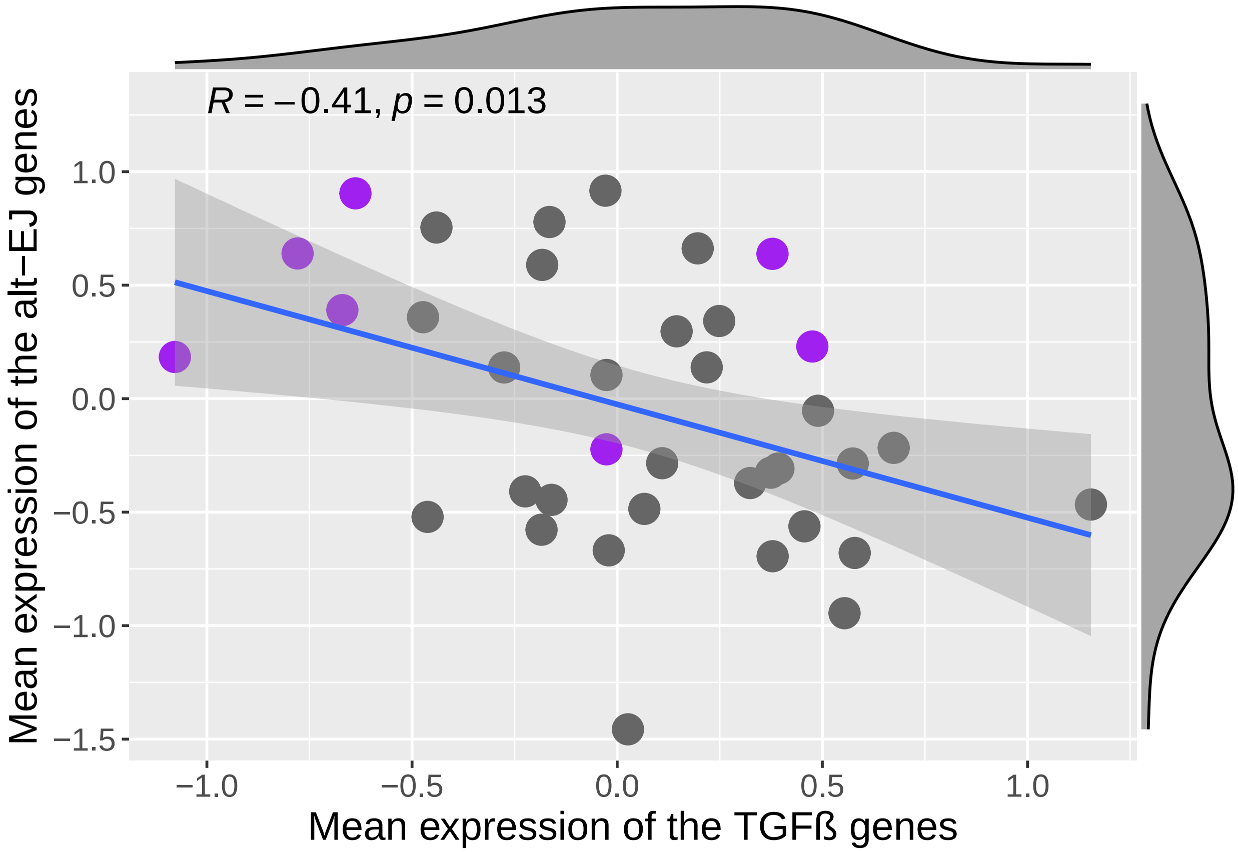
#3. Scatterplot of ALTEJ vs 53BP1 with HPV color.

```
p <- ggplot(DF, aes(x=Mean_ALTEJ, y=`5Gy_% (>5 foci)`)) +
  geom_point(size=5, aes(color=HPV_Status)) +
  scale_color_manual(values=c("grey40", "purple", "grey40")) +
  labs(x = "Mean expression of the alt-EJ genes", y = "% 53BP1+ cells", col="HPV status") +
  geom_smooth(method="glm", fullrange=TRUE) + #Add correlation line
  stat_cor(label.x=-1.4, label.y=30, method="spearman", size=5) +
  theme(text = element_text(size=15)) +
  theme(legend.title=element_text(size=8))
p <- p + rremove("legend")
p1 <- ggMarginal(p,  size=10, fill = "grey65"); p1
```



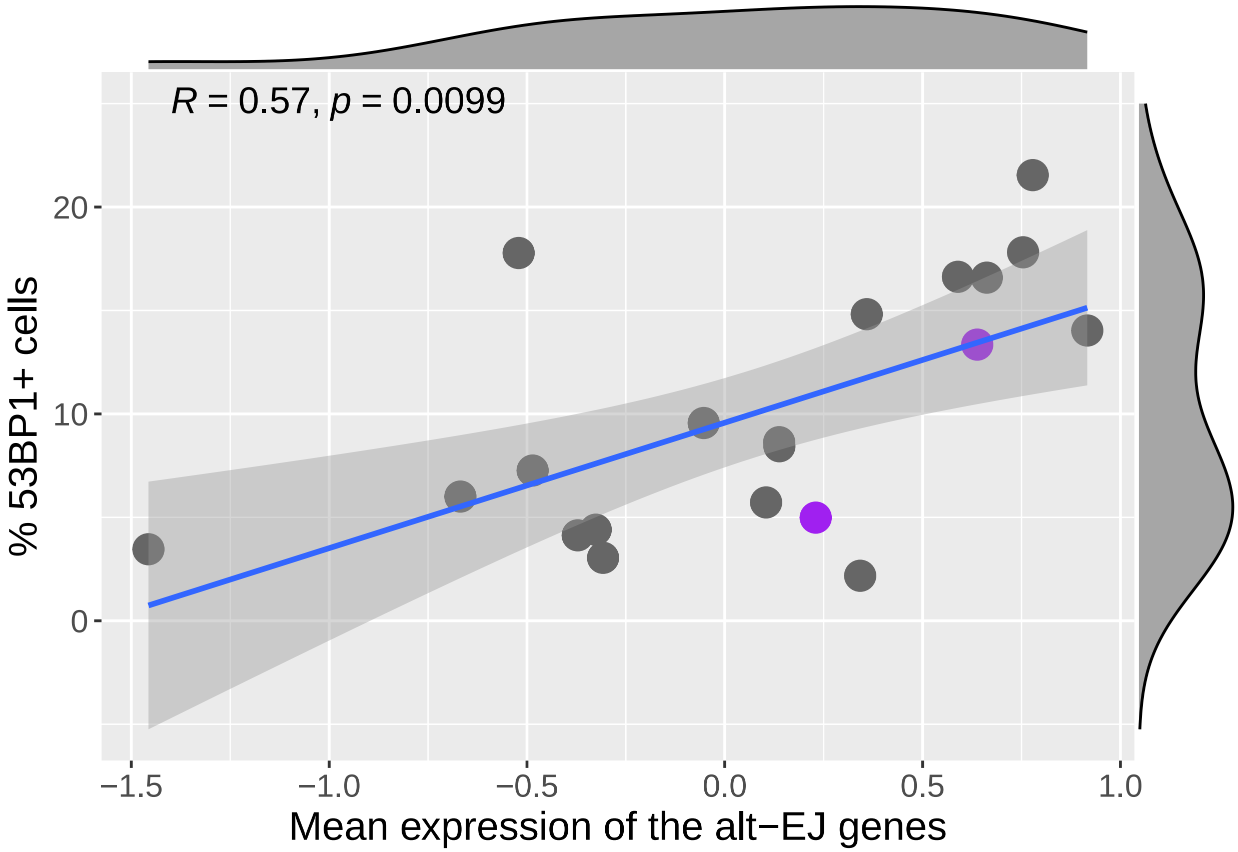
#4. Scatterplot of TGFB vs ALTEJ with HPV color.

```
p <- ggplot(DF, aes(x=Mean_TGFB, y=Mean_ALTEJ)) +
  geom_point(size=5, aes(color=HPV_Status)) +
  scale_color_manual(values=c("grey40", "purple", "grey40")) +
  labs(x = "Mean expression of the TGF  genes", y = "Mean expression of the alt-EJ genes", col="HPV stat
  geom_smooth(method="glm", fullrange=TRUE) + #Add correlation line
  stat_cor(label.x=-1, label.y=1.3, method="spearman", size=5) +
  theme(text = element_text(size=15)) +
  theme(legend.title=element_text(size=8))
p <- p + rremove("legend")
p1 <- ggMarginal(p,  size=10, fill = "grey65"); p1
```



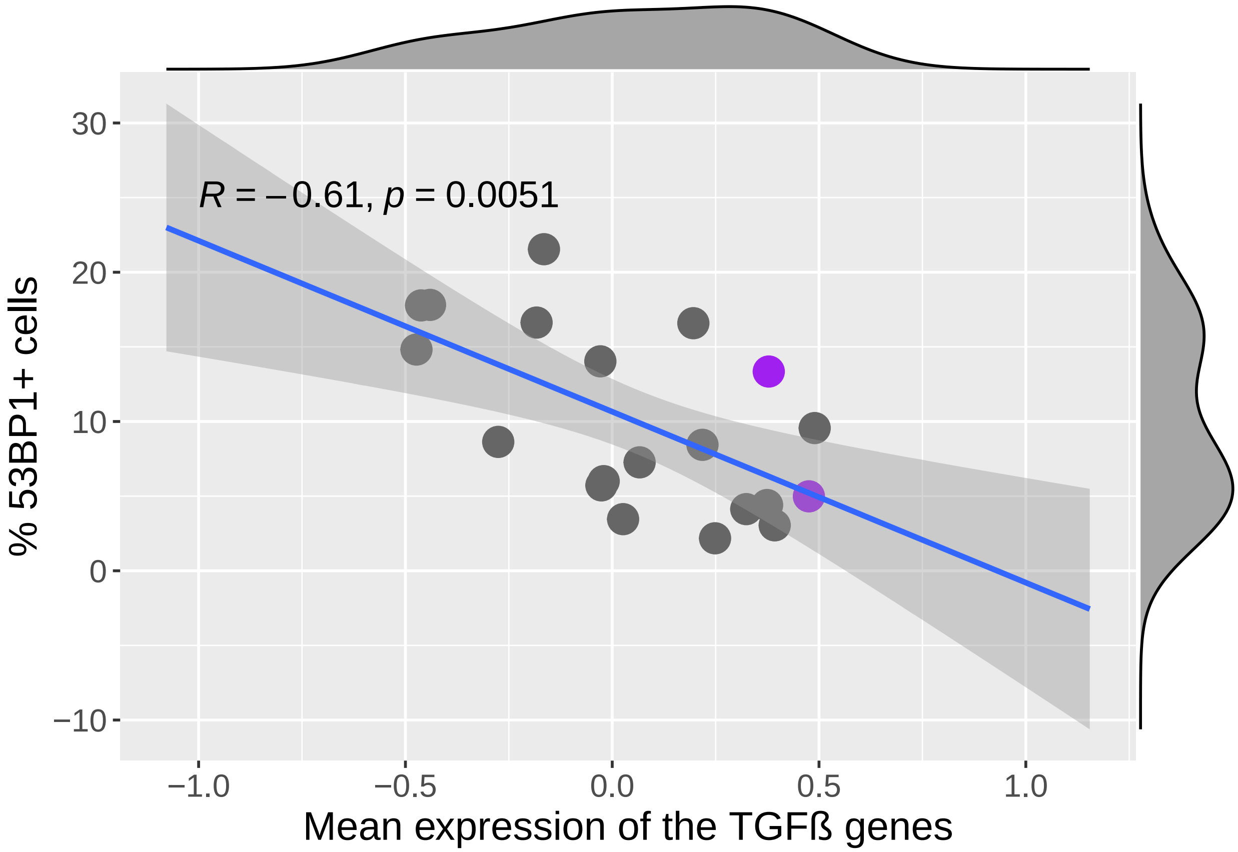
#5. Scatterplot of ALTEJ vs 53BP1 (after treatment with olaparib) with HPV color.

```
p <- ggplot(DF, aes(x=Mean_ALTEJ, y=`Olaparib_% (>5 foci)`)) +
  geom_point(size=5, aes(color=HPV_Status)) +
  scale_color_manual(values=c("grey40", "purple", "grey40")) +
  labs(x = "Mean expression of the alt-EJ genes", y = "% 53BP1+ cells", col="HPV status") +
  geom_smooth(method="glm", fullrange=TRUE) + #Add correlation line
  stat_cor(label.x=-1.4, label.y=25, method="spearman", size=5) +
  theme(text = element_text(size=15)) +
  theme(legend.title=element_text(size=8))
p <- p + rremove("legend")
p1 <- ggMarginal(p,  size=10, fill = "grey65"); p1
```



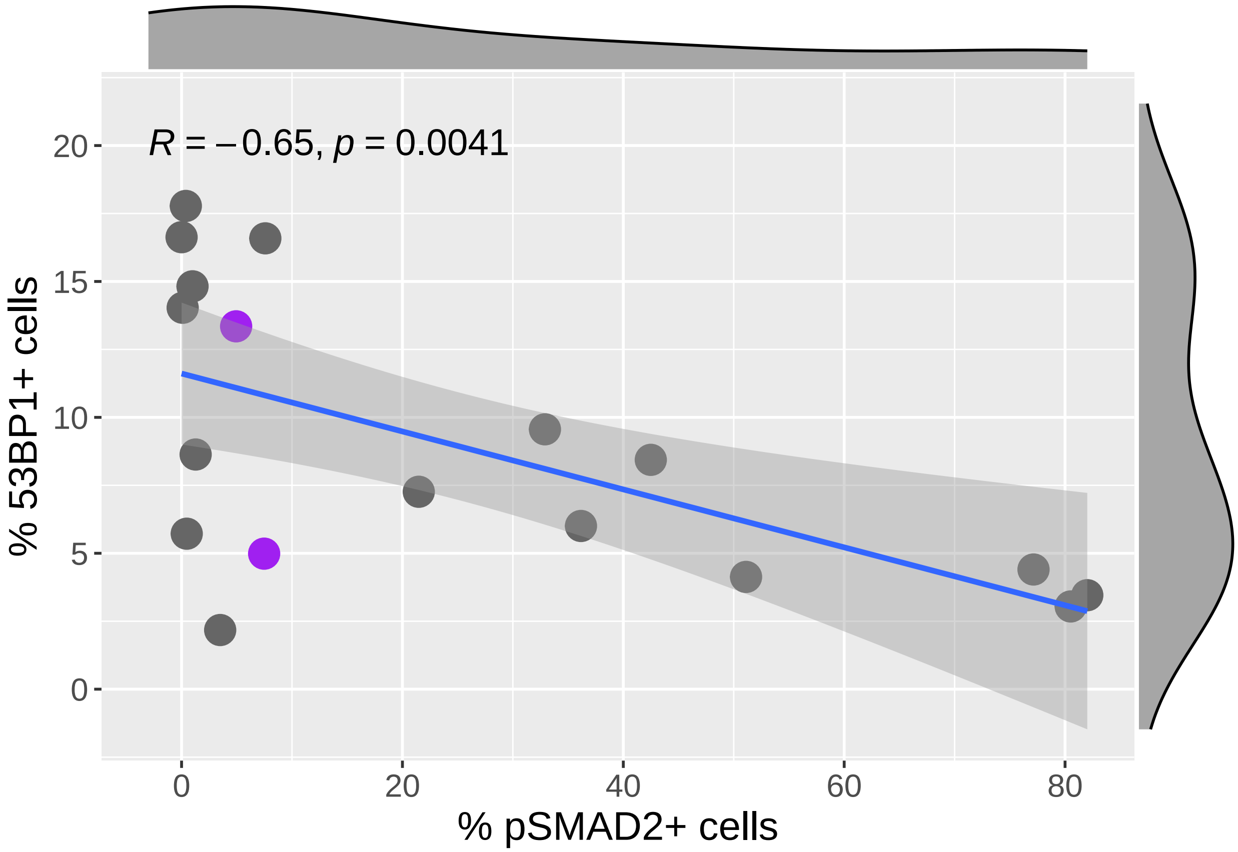
#6. Scatterplot of TGFB vs 53BP1 (after treatment with olaparib) with HPV color.

```
p <- ggplot(DF, aes(x=Mean_TGFB, y=`Olaparib_% (>5 foci)`)) +
  geom_point(size=5, aes(color=HPV_Status)) +
  scale_color_manual(values=c("grey40", "purple", "grey40")) +
  labs(x = "Mean expression of the TGF  genes", y = "% 53BP1+ cells", col="HPV status") +
  geom_smooth(method="glm", fullrange=TRUE) + #Add correlation line
  stat_cor(label.x=-1, label.y=25, method="spearman", size=5) +
  theme(text = element_text(size=15)) +
  theme(legend.title=element_text(size=8))
p <- p + rremove("legend")
p1 <- ggMarginal(p,  size=10, fill = "grey65"); p1
```



#7. Scatterplot of PSMAD2 vs PARPi 53BP1 with HPV color.

```
p <- ggplot(DF, aes(x=`pSMAD2_%`, y=`Olaparib_% (>5 foci)`)) +
  geom_point(size=5, aes(color=HPV_Status)) +
  scale_color_manual(values=c("grey40", "purple", "grey40")) +
  labs(x = "% pSMAD2+ cells", y = "% 53BP1+ cells", col="HPV status") +
  geom_smooth(method="glm", fullrange=TRUE) +
  stat_cor(label.x=-3, label.y=20, method="spearman", size=5) +
  theme(text = element_text(size=15)) +
  theme(legend.title=element_text(size=8))
p <- p + rremove("legend")
p1 <- ggMarginal(p,  size=10, fill = "grey65"); p1
```



PART 6.A: VOLCANO PLOTS - GENES' ASSOCIATION WITH PSMAD2

#1. Create a dataframe with the expression of each gene and 53BP1 & pSmad2.

```
Miniexp <- Exp[which(Exp$Signature.x == "TGFB" | Exp$Signature.x == "ALTEJ"),] #86 genes.
table(Miniexp$Signature.x) #50 TGFB genes, 36 ALTEJ genes.
rownames(Miniexp) <- Miniexp$...1
Miniexp1 <- as.matrix(Miniexp[,-(1:2)])
rownames(Miniexp1) <- rownames(Miniexp)
Miniexp <- Miniexp1
class(Miniexp)<-"numeric" #turn the values into numeric
DF1 <- data.frame(t(Miniexp))
DF1$Nanostring_ID <- rownames(DF1)
DF1 <- merge(DF, DF1, by.x="Nanostring_ID", by.y="Nanostring_ID", all.x=TRUE, all.y=TRUE)
```

#2. Run a Spearman Correlation analysis on multiple features (the expression of each ALTEJ and TGFB gene) versus PSMAD2.

```
Bothvector <- unlist(Bothgenelists)
names(DF1)
Allgenesvector <- append(Bothvector, "pSMAD2_%")
A <- apply(DF1[,which(colnames(DF1) %in% Allgenesvector)], 2, cor.test, DF1$`pSMAD2_%`, method="spearma
```

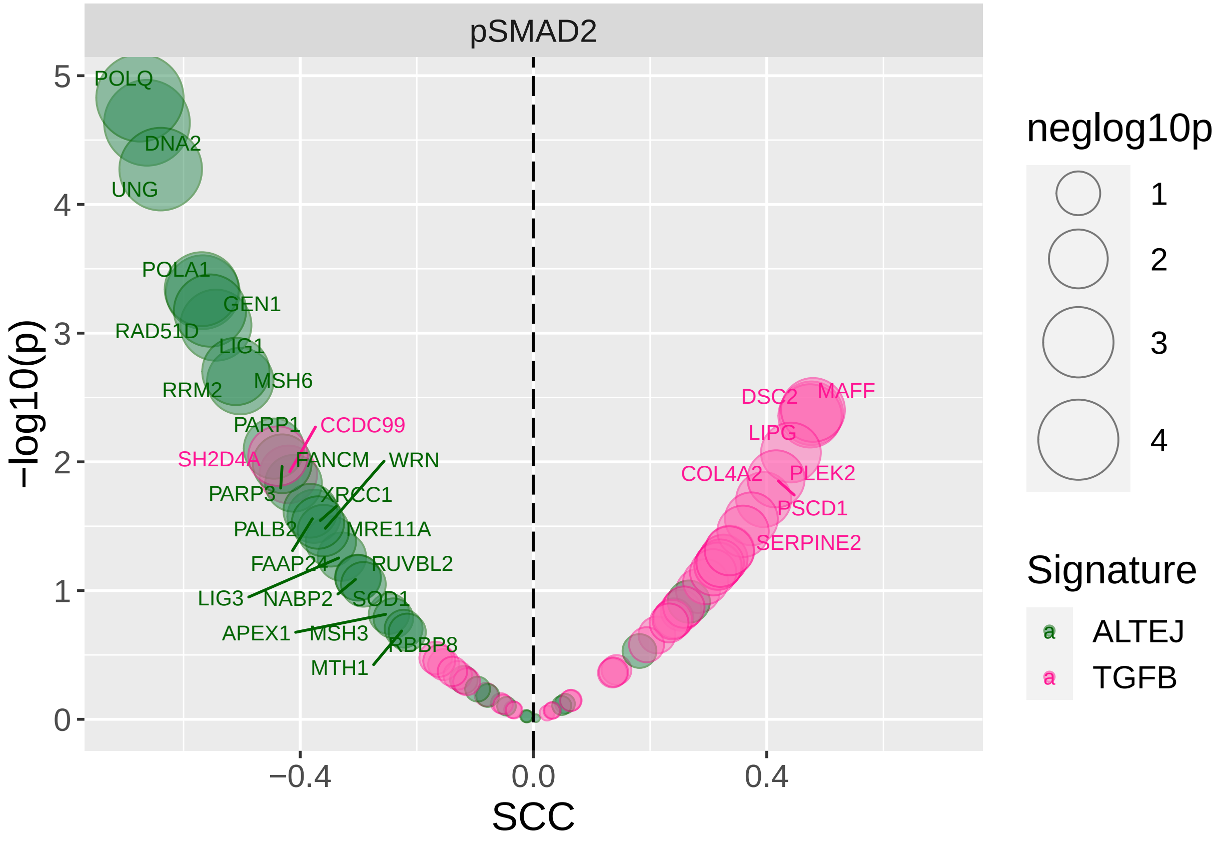
```r
class(A) #list
B1 <- data.frame(sapply(A, "[[", "p.value")) #Extract elements from the nested list.
B2 <- data.frame(sapply(A, "[[", "estimate"))
B2$Gene <-substring(rownames(B2),1,nchar(rownames(B2))-4) #Exclude the last 4 characters
B1$Gene <- rownames(B1)
B <- merge(B1, B2, by.x="Gene", by.y="Gene")
B$p.value <- B$sapply.A.........p.value..
B$SCC <- B$sapply.A.........estimate..
B$neglog10p <- -log(B$p.value, base = 10)
rownames(B) <- B$Gene
B$sapply.A.........p.value.. <- NULL
B$sapply.A.........estimate.. <- NULL
##Check that the results are correct.
cor.test(DF1$APE2, DF1$`pSMAD2_%`, method = "spearman")
```

#3. Create a new variable that is the signature.

```r
B$Signature <- ifelse(B$Gene %in% TGFBlist, "TGFB", "ALTEJ")
```

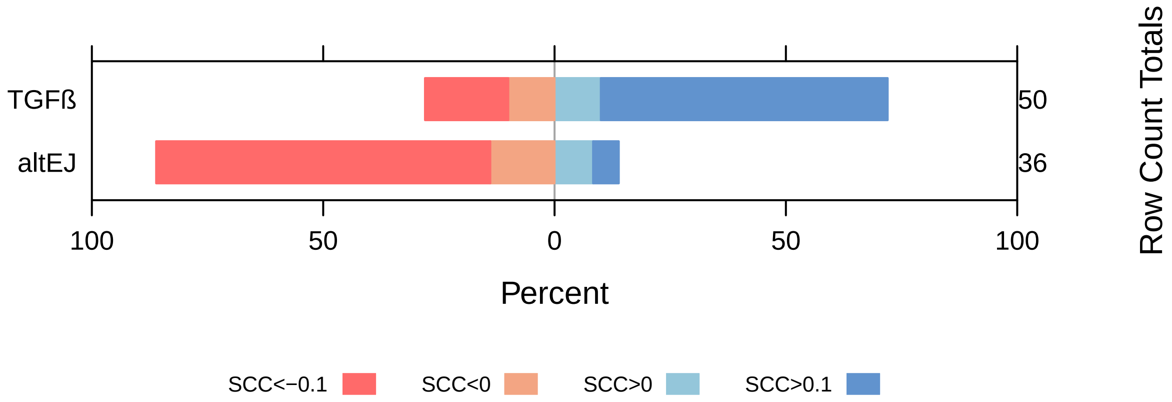
#4. Create a volcano plot.

```r
B$title <- "pSMAD2"
ggplot(B[-which(B$Gene=="pSMAD2_%"),], aes(x=SCC, y=neglog10p)) +
  geom_point(alpha=0.5, shape = 21, aes(size=neglog10p, col=Signature, fill=Signature)) +
  labs(x = "SCC", y = "-log10(p)") +
  geom_vline(xintercept = 0, linetype="longdash")+
  #geom_hline(yintercept = 1.3, linetype="longdash") +
  geom_text_repel(aes(SCC, neglog10p, label=Gene, col=Signature), size=2.7)+
  xlim(-0.7, 0.7) + ylim(0, 4.9) +
  theme(text = element_text(size=15)) +
  facet_grid(. ~ title, scales="free_x") +
  scale_size(range = c(1, 15)) +
  scale_fill_manual(values=c("seagreen4", "hotpink")) +
  scale_color_manual(values=c("darkgreen", "deeppink")) #PDF 7x8.5
```



#5. Represent the number of genes in each direction.

```
B<-B[-which(B$Gene=="pSMAD2_%"),]
B$Signature <- ifelse(B$Gene %in% TGFBlist, "TGF ", "altEJ")
B$direction <- ifelse(B$SCC < -0.1, "SCC<-0.1",
                      ifelse(B$SCC > 0.1, "SCC>0.1",
                      ifelse(B$SCC > 0 & B$SCC < 0.1, "SCC>0",
                      ifelse(B$SCC > -0.1 & B$SCC < 0, "SCC<0", NA))))
counts <- data.frame(table(B$Signature, B$direction))
counts$Signature <- counts$Var1
counts$Direction <- counts$Var2
counts$Var1 <- NULL
counts$Var2 <- NULL
data_l <- spread(counts, key = Direction, value = Freq)
row.names(data_l) <- data_l$Signature
data_l$Signature<- NULL
data_l <- data_l[c(2,1),]
library("HH")
HH::likert(data_l, horizontal=TRUE,aspect=0.15,
           as.percent=TRUE,
           col=c("indianred1", "#f3a583", "#94c6da", "#6193CE"),
           main="",
           xlim=c(-100,100)) #PDF 5x8.5
```



#6. Rank the genes by their correlation with pSMAD2.

```
TGFBPSMAD2 <- B[which(B$Gene %in% TGFBlist), ]
TGFBPSMAD2 <- TGFBPSMAD2 %>% mutate(rank_PSMAD2 = dense_rank(desc(SCC)))
ALTEJPSMAD2 <- B[which(B$Gene %in% ALTEJlist), ]
ALTEJPSMAD2 <- ALTEJPSMAD2 %>% mutate(rank_PSMAD2 = dense_rank(SCC))
```

PART 6.B: VOLCANO PLOTS - GENES’ ASSOCIATION WITH 53BP1

#2. Run Spearman Correlation analysis on multiple features (the expression of each ALTEJ and TGFB gene) versus 53BP1.

```
Bothvector <- unlist(Bothgenelists)
names(DF1)
Allgenesvector <- append(Bothvector, "5Gy_% (>5 foci)%")
Allgenesvector
A <- apply(DF1[,which(colnames(DF1) %in% Allgenesvector)], 2, cor.test, DF1$`5Gy_% (>5 foci)`, method="
class(A) #list
B1 <- data.frame(sapply(A, "[[", "p.value")) #Extract elements from the nested list.
B2 <- data.frame(sapply(A, "[[", "estimate"))
B2$Gene <-substring(rownames(B2),1,nchar(rownames(B2))-4) #Exclude the last 4 characters
B1$Gene <- rownames(B1)
B <- merge(B1, B2, by.x="Gene", by.y="Gene")
```

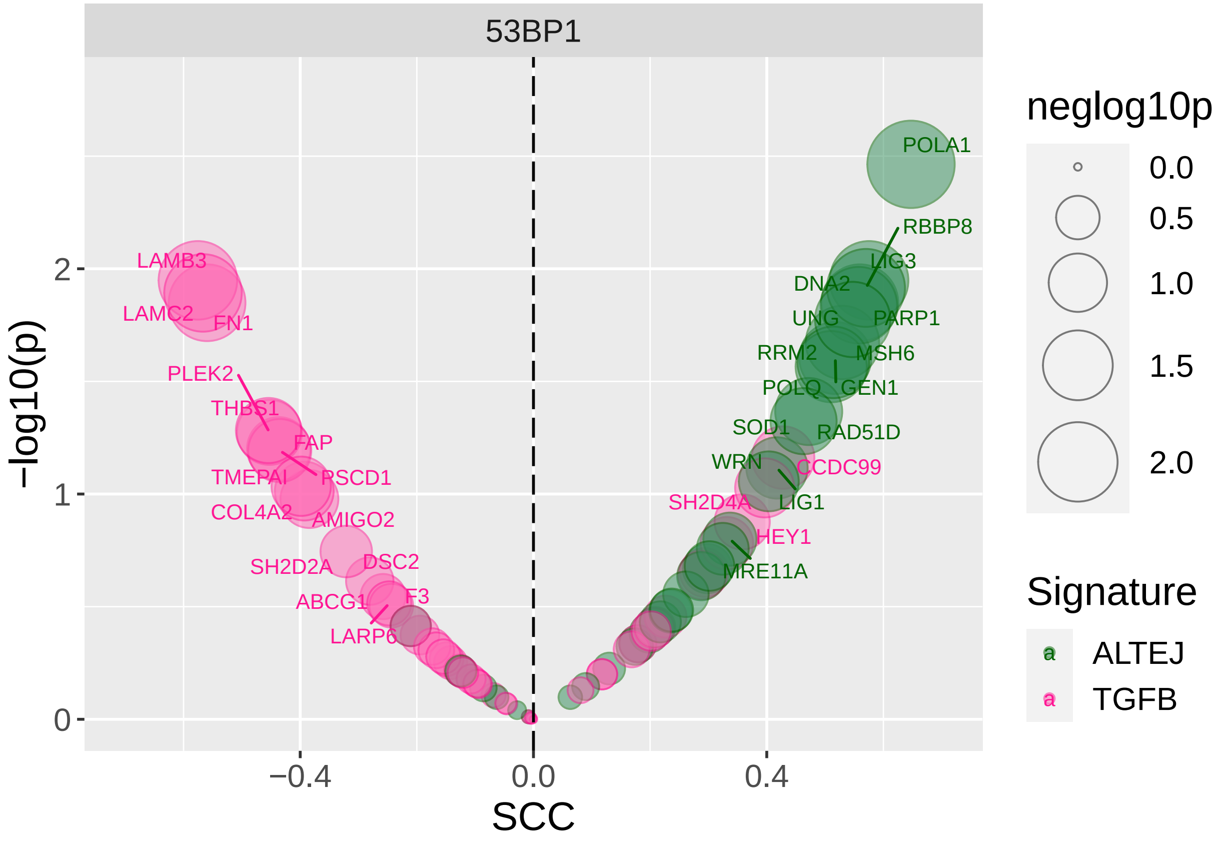
```
B$p.value <- B$sapply.A.........p.value..
B$SCC <- B$sapply.A.........estimate..
B$neglog10p <- -log(B$p.value, base = 10)
rownames(B) <- B$Gene
B$sapply.A.........p.value.. <- NULL
B$sapply.A.........estimate.. <- NULL
##Check that the results are correct.
cor.test(DF1$APE2, DF1$`5Gy_% (>5 foci)`, method = "spearman")
```

#3. Create a new variable that is the signature.

```
B$Signature <- ifelse(B$Gene %in% TGFBlist, "TGFB", "ALTEJ")
table(B$Signature)
```

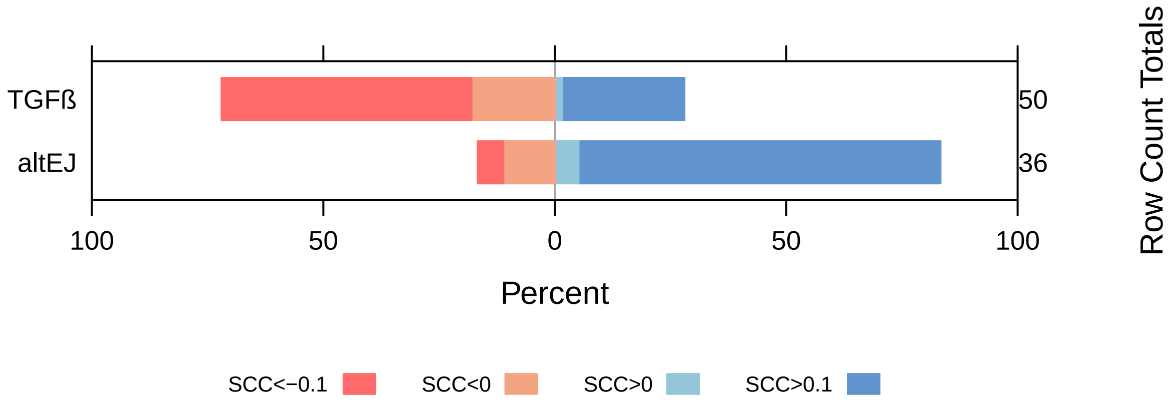
#4. Create a volcano plot.

```
B$title <- "53BP1"
ggplot(B, aes(x=SCC, y=neglog10p)) +
  geom_point(alpha=0.5, shape = 21, aes(size=neglog10p, col=Signature, fill=Signature)) +
  labs(x = "SCC", y = "-log10(p)") +
  geom_vline(xintercept = 0, linetype="longdash") +
  #geom_hline(yintercept = 1.3, linetype="longdash") +
  geom_text_repel(aes(SCC, neglog10p, label=Gene, col=Signature), size=2.7)+
  xlim(-0.7, 0.7) + ylim(0, 2.8) +
  theme(text = element_text(size=15)) +
  facet_grid(. ~ title, scales="free_x") +
  scale_size(range = c(1, 15)) +
  scale_fill_manual(values=c("seagreen4", "hotpink")) +
  scale_color_manual(values=c("darkgreen", "deeppink")) #PDF 7x8.5
```



#5. Represent the number of genes in each direction.

```
B$Signature <- ifelse(B$Gene %in% TGFBlist, "TGF ", "altEJ")
table(B$Signature)
B$direction <- ifelse(B$SCC < -0.0999999, "SCC<-0.1",
                      ifelse(B$SCC > 0.1, "SCC>0.1",
                      ifelse(B$SCC > 0 & B$SCC < 0.1, "SCC>0",
                      ifelse(B$SCC > -0.1 & B$SCC < 0.00000001, "SCC<0", NA))))
counts <- data.frame(table(B$Signature, B$direction))
counts$Signature <- counts$Var1
counts$Direction <- counts$Var2
counts$Var1 <- NULL
counts$Var2 <- NULL
data_l <- spread(counts, key = Direction, value = Freq)
row.names(data_l) <- data_l$Signature
data_l$Signature<- NULL
data_l <- data_l[c(2,1),]
library("HH")
HH::likert(data_l, horizontal=TRUE,aspect=0.15,
           as.percent=TRUE,
           col=c("indianred1", "#f3a583", "#94c6da", "#6193CE"),
           main="",
           xlim=c(-100,100)) #PDF 5x8.5
```



#4. Rank the genes by their correlation with 53BP1.

```
TGFB53BP1 <- B[which(B$Gene %in% TGFBlist), ]
TGFB53BP1 <- TGFB53BP1 %>% mutate(rank_53BP1 = dense_rank(SCC))
ALTEJ53BP1 <- B[which(B$Gene %in% ALTEJlist), ]
ALTEJ53BP1 <- ALTEJ53BP1 %>% mutate(rank_53BP1 = dense_rank(desc(SCC)))
```

PART 7: GENE CORRELATION MATRIX

#1. Create a dataframe with the expression of TGFB and ALTEJ genes. Rows=samples, columns=genes.

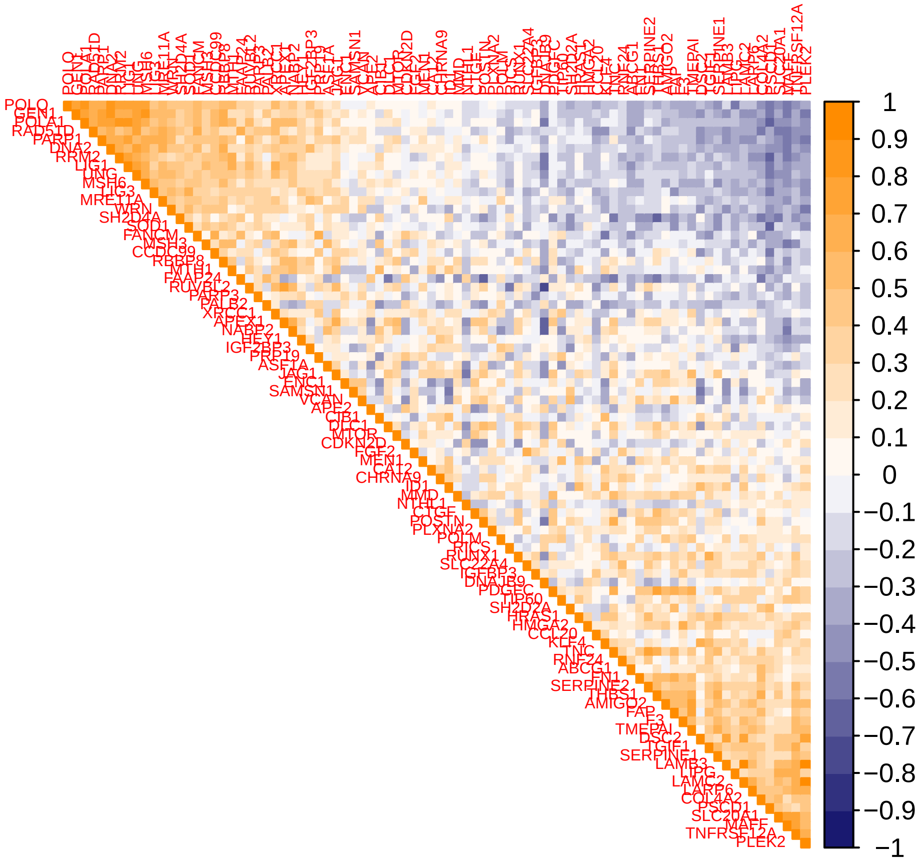
```
Miniexp <- Exp[which(Exp$Signature.x == "TGFB" | Exp$Signature.x == "ALTEJ"),] #86g.
rownames(Miniexp) <- Miniexp$...1
Miniexp1 <- Miniexp[,-(1:2)] #Remove signature and gene_ID columns.
rownames(Miniexp1) <- rownames(Miniexp)
DF1 <- as.data.frame(t(Miniexp1))
DF1 <- data.frame(sapply(DF1, function(x) as.numeric(as.character(x)))) #Convert the values into numeri
```

#2. Create a matrix with the Pearson correlation coefficient between each pair of genes.

```
GeneCorr <- cor(DF1, method = c("pearson"))
cor.test(DF1$ABCG1, DF1$AMIGO2, method="pearson") #Check that the results are correct.
class(GeneCorr) #[1] "matrix"
```

#3. Plot a correlation matrix.

```
corrplot(GeneCorr, type = "upper", method = "color", order = "FPC", tl.cex = 0.5,
         col = colorRampPalette(c("midnightblue","white","darkorange"))(20))
```

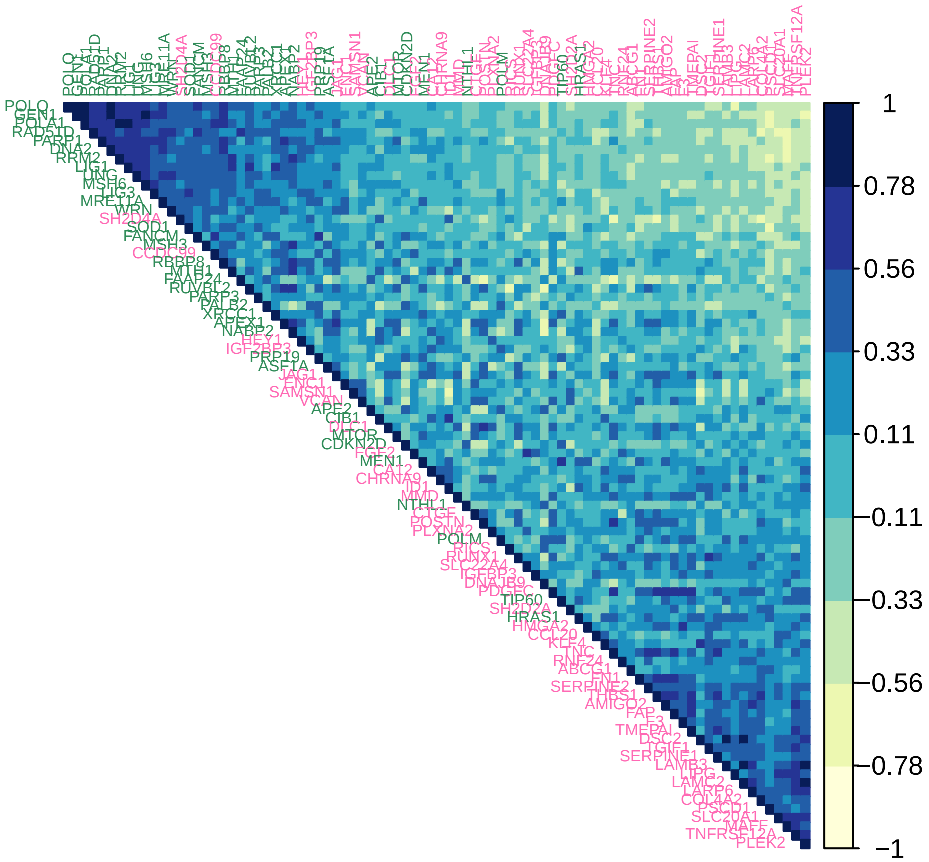


#4. Assign colors to the genes according to their signature.

```
##Signature is a dataframe with "rownames" the genes and a variable "Signature".
GeneColors <- ifelse(Signature$Signature == "TGFB", "hotpink", "seagreen4")
names(GeneColors) <- names(Signature)
##Order the colors like the order or the genes in the correlation matrix.
Order <- corrMatOrder(GeneCorr, order="FPC")
GeneColors <- GeneColors[Order]
```

#5. Plot again the correlation matrix but adding the colors to the genes.

```
corrplot(GeneCorr, type = "upper", method = "color", order = "FPC", tl.cex = 0.52, #Type="upper" or "fu
         tl.col = GeneColors,
         #col = colorRampPalette(col=c("indianred1","white","steelblue4"))(20))
         col = brewer.pal(9,"YlGnBu"))
```



PART 8: WEIGHTED GENE COEXPRESSION NETWORK —

#1.Create a dataframe with the expression of TGFB and ALTEJ genes. Rows=samples, columns=genes.

```
Miniexp <- Exp[which(Exp$Signature.x == "TGFB" | Exp$Signature.x == "ALTEJ"),] #86g.
rownames(Miniexp) <- Miniexp$...1
Miniexp1 <- Miniexp[,-(1:2)] #Remove signature and gene_ID columns.
rownames(Miniexp1) <- rownames(Miniexp)
DF1 <- as.data.frame(t(Miniexp1))
DF1 <- data.frame(sapply(DF1, function(x) as.numeric(as.character(x)))) #Convert the values into numeri
```

#2. Create a dataframe with the Pearson correlation coefficient between each pair of genes.

```
GeneCorr <- DF1 %>% correlate() %>% stretch()
cor.test(DF1$ABCG1, DF1$AMIGO2, method="pearson") #Check that the results are correct.
```

#2.Create a tbl_graph object.

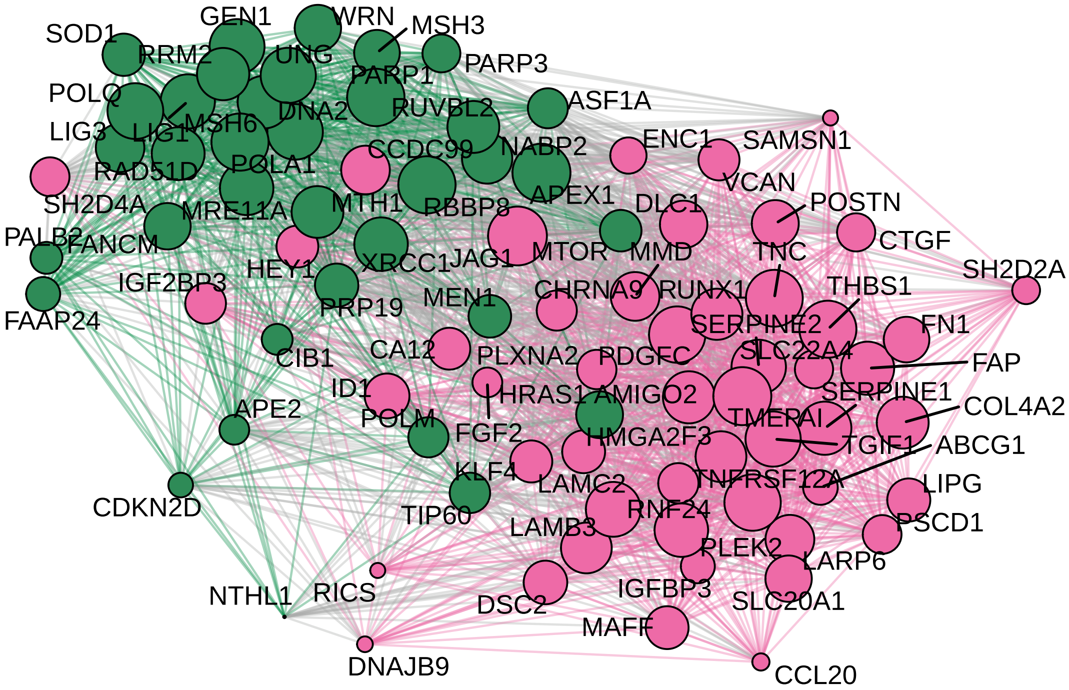
```
##Create a dataframe with the Edges.
Edges <- GeneCorr %>% filter(r > 0.007)
Edges$class <- "Mixed edge"
Edges <- within(Edges, class[Edges$x %in% TGFBlist & Edges$y %in% TGFBlist] <- "TGFB edge")
Edges <- within(Edges, class[Edges$x %in% ALTEJlist & Edges$y %in% ALTEJlist] <- "ALTEJ edge")
```

```r
##Create a dataframe with the Nodes.
Nodes <- data.frame(Gene=colnames(DF1))
Nodes$signature <- ifelse(Nodes$Gene %in% TGFBlist, "TGFB signature",
                          ifelse(Nodes$Gene %in% ALTEJlist, "ALTEJ signature", NA))
rownames(Nodes) <- Nodes$Gene
```

```r
##Create the tbl_graph object.
Tbl_graph1 <- tbl_graph(nodes = Nodes, edges = Edges, directed = FALSE)
```

#3. Plot the weighted gene coexpression network.

```r
ggraph(Tbl_graph1, layout = "fr", weights = r) +
  geom_edge_link2(aes(colour=class), edge_alpha=0.2, edge_width=0.4) +
  geom_node_point(aes(fill=signature, size=centrality_degree(weights=r)), shape=21) +
  geom_node_text(aes(label = Gene), colour="black", size=3.5, repel = TRUE) +
  scale_edge_color_manual(values = c("springgreen4", "grey68", "hotpink2")) +
  scale_fill_manual(values = c("seagreen4", "hotpink2")) +
  scale_size(range = c(0.1, 10)) +
  theme_graph() + rremove("legend") #PDF7x8
```



PART 9: CALCULATE GENES' CENTRALITY DEGREE

#1. Calculate the weighted centrality degree of each node (gene) within its own signature. #Note: The weighted centrality degree depends on the number of edges connecting a node and their weight.

```
##TGFB centrality of TGFB genes
EdgesTGFB <- Edges %>% filter(class == c("TGFB edge"))
Tbl_graphTGFB <- tbl_graph(nodes = Nodes, edges = EdgesTGFB, directed = FALSE)
Tbl_graphTGFB <- Tbl_graphTGFB %>% mutate(centrality_TGFB_TGFBg=centrality_degree(weights=r))
CentralityTGFB_TGFBg <- Tbl_graphTGFB %>%
  activate(nodes) %>%
  as_tibble() %>%
  arrange(desc(centrality_TGFB_TGFBg))
```

```
##TGFB centrality of ALTEJ genes and ALTEJ centrality of TGFB genes.
EdgesMixed <- Edges %>% filter(class == c("Mixed edge"))
Tbl_graphMixed <- tbl_graph(nodes = Nodes, edges = EdgesMixed, directed = FALSE)
Tbl_graphMixed <- Tbl_graphMixed %>% mutate(centrality_Mixed=centrality_degree(weights=r))
CentralityMixed <- Tbl_graphMixed %>%
  activate(nodes) %>%
  as_tibble() %>%
  arrange(desc(centrality_Mixed))
```

```
##ALTEJ centrality of ALTEJ genes.
EdgesALTEJ <- Edges %>% filter(class == c("ALTEJ edge"))
Tbl_graphALTEJ <- tbl_graph(nodes = Nodes, edges = EdgesALTEJ, directed = FALSE)
Tbl_graphALTEJ <- Tbl_graphALTEJ %>% mutate(centrality_ALTEJ_ALTEJg=centrality_degree(weights=r))
CentralityALTEJ_ALTEJg <- Tbl_graphALTEJ %>%
  activate(nodes) %>%
  as_tibble() %>%
  arrange(desc(centrality_ALTEJ_ALTEJg))
```

#2. Merge all the calculated centralities in one single dataframe.

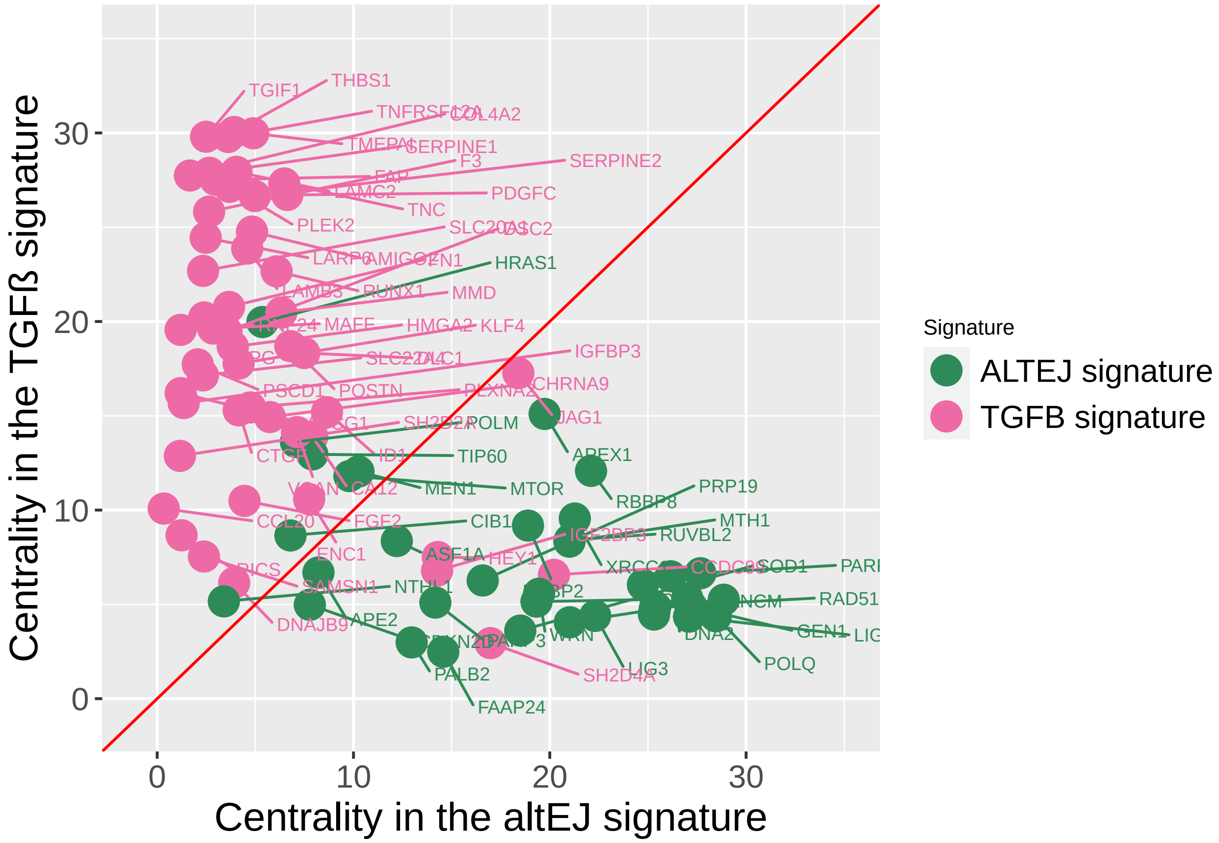
```
Centrality <- merge(CentralityTGFB_TGFBg, CentralityALTEJ_ALTEJg, by.x="Gene", by.y="Gene",  all.x=TRUE
Centrality <- merge(Centrality, CentralityMixed, by.x="Gene", by.y="Gene",  all.x=TRUE, all.y=TRUE)
```

#3. Calculate the centrality of each gene within both signatures.

```
Centrality$centrality_TGFB_new <- NA
Centrality$centrality_TGFB_new <- ifelse(Centrality$signature =="TGFB signature", Centrality$centrality_
Centrality$centrality_ALTEJ_new <- NA
Centrality$centrality_ALTEJ_new <- ifelse(Centrality$signature =="ALTEJ signature", Centrality$centrali
Centrality <- Centrality[,c("Gene", "signature", "centrality_TGFB_new", "centrality_ALTEJ_new")]
```

#4. Plot the weighted centrality degree of the genes in both signatures.

```
ggplot(Centrality, aes(x=centrality_ALTEJ_new, y=centrality_TGFB_new, col=signature)) + geom_point(size=
  scale_color_manual(values=c("seagreen4", "hotpink2"), na.translate=TRUE, na.value="grey") +
  labs(x = "Centrality in the altEJ signature", y = "Centrality in the TGF  signature", color = "Signatu
  theme(legend.title=element_text(size=8)) + theme(text = element_text(size=15)) +
  geom_text_repel(aes(label=Centrality$Gene, col=Centrality$signature), hjust=2, vjust=2, size=2.5) +
  geom_abline(intercept = 0, slope = 1, colour = "red") +
  xlim(-1, 35) + ylim(-1, 35)
```



#5. Rank the genes by their weighted centrality degree within their signature network.

```
TGFBcentrality <- Centrality[which(Centrality$Gene %in% TGFBlist), ]
TGFBcentrality <- TGFBcentrality %>% mutate(rank_centrality = dense_rank(desc(centrality_TGFB_new)))
ALTEJcentrality <- Centrality[which(Centrality$Gene %in% ALTEJlist), ]
ALTEJcentrality <- ALTEJcentrality %>% mutate(rank_centrality = dense_rank(desc(centrality_ALTEJ_new)))
```

PART 10: WEIGHT GENES' IMPORTANCE (BASED ON THE RESULTS FROM PARTS 6 & 9)

#1. Merge the results of parts 6 & 9 into two files.

```
##TGFB
TGFBimportance<-merge(TGFB53BP1, TGFBPSMAD2, by.x = "Gene", by.y = "Gene", all.x=TRUE, all.y=TRUE)
TGFBimportance<-merge(TGFBimportance, TGFBcentrality, by.x = "Gene", by.y = "Gene", all.x=TRUE, all.y=T
##ALTEJ
ALTEJimportance<-merge(ALTEJ53BP1, ALTEJPSMAD2, by.x = "Gene", by.y = "Gene", all.x=TRUE, all.y=TRUE)
ALTEJimportance<-merge(ALTEJimportance, ALTEJcentrality, by.x = "Gene", by.y = "Gene", all.x=TRUE, all.
```

#2. Calculate the mean rank of each gene.

```
TGFBimportance$Mean_rank <- rowMeans(TGFBimportance[,c("rank_53BP1", "rank_PSMAD2", "rank_centrality")]
ALTEJimportance$Mean_rank <- rowMeans(ALTEJimportance[,c("rank_53BP1", "rank_PSMAD2", "rank_centrality"
```

#3. Rescale the values of the weighted centrality degrees to a 0-0.5 range.

```
library("scales")
TGFBimportance$centrality_TGFB_scaled <- rescale(TGFBimportance$centrality_TGFB_new, to = c(0, 0.5))
ALTEJimportance$centrality_ALTEJ_scaled <- rescale(ALTEJimportance$centrality_ALTEJ_new, to = c(0, 0.5)
```

#4. Calculate the weight of each gene and tidy the dataframes.

```
##TGFB: A weight for each TGFB signature gene will be assigned based on the average of its:
  ##Spearman correlation coeffcient with the % of pSmad2 positive cells.
  ##-1 * Spearman correlation coeffcient with the % of 53BP1 positive cells (after 5Gy irradiation).
  ##Weighted centrality degree within the TGFB signature, rescaled to a 0-0.5 range.
TGFBimportance$pSmad2 <- TGFBimportance$SCC.y
TGFBimportance$`53BP1` <- TGFBimportance$SCC.x * -1
TGFBimportance$centrality <- TGFBimportance$centrality_TGFB_scaled
TGFBimportance$`mean weight` <- rowMeans(TGFBimportance[,c("pSmad2", "53BP1", "centrality")], na.rm=TRU
TGFBimportance$readme <- "pSmad2 = Spearman correlation coeffcient with the % of pSmad2 positive cells;
53BP1 = -1 * Spearman correlation coeffcient with the % of 53BP1 positive cells (after 5Gy irradiation)
centrality = Centrality degree within the TGFB signature, rescaled to a 0-0.5 range;
mean weight = average of the former three columns"
TGFBimportance <- TGFBimportance[,c("Gene","pSmad2", "53BP1", "centrality", "mean weight", "readme",
                                     "rank_PSMAD2", "rank_53BP1", "rank_centrality", "Mean_rank")]
```

```
##ALTEJ: A weight for each ALTEJ signature gene will be assigned based on the average of its:
  ##-1* Spearman correlation coeffcient with the % of pSmad2 positive cells.
  ##Spearman correlation coeffcient with the % of 53BP1 positive cells (after 5Gy irradiation).
  ##Weighted centrality degree within the ALTEJ signature, rescaled to a 0-0.5 range.
ALTEJimportance$pSmad2 <- ALTEJimportance$SCC.y * -1
ALTEJimportance$`53BP1` <- ALTEJimportance$SCC.x
ALTEJimportance$centrality <- ALTEJimportance$centrality_ALTEJ_scaled
ALTEJimportance$`mean weight` <- rowMeans(ALTEJimportance[,c("pSmad2", "53BP1", "centrality")], na.rm=T
ALTEJimportance$readme <- "pSmad2 = -1 * Spearman correlation coeffcient with the % of pSmad2 positive 
53BP1 = Spearman correlation coeffcient with the % of 53BP1 positive cells (after 5Gy irradiation);
centrality = Centrality degree within the ALTEJ signature, rescaled to a 0-0.5 range;
mean weight = average of the former three columns"
ALTEJimportance <- ALTEJimportance[,c("Gene","pSmad2", "53BP1", "centrality", "mean weight", "readme",
                                     "rank_PSMAD2", "rank_53BP1", "rank_centrality", "Mean_rank")]
```

#5. Export the results.

```
write.xlsx(TGFBimportance, file="Output/Relative importance of BAlt genes.xlsx", col.names = TRUE, row.
write.xlsx(ALTEJimportance, file="Output/Relative importance of BAlt genes.xlsx", col.names = TRUE, row
```

PART 11: 3D SCATTERPLOTS OF GENES' IMPORTANCE —

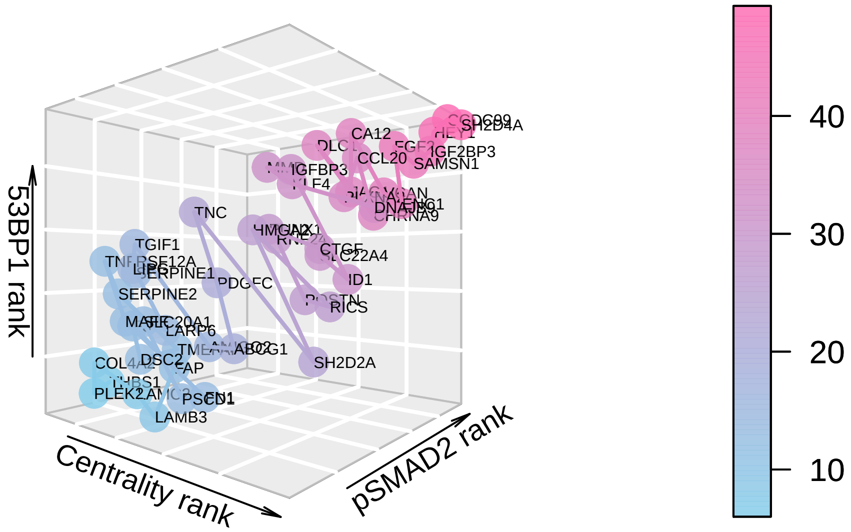
#1. Do a 3D plot of genes' importance with PLOTLY.

```r
##TGFB genes
TGFBimportance2 <- TGFBimportance[order(TGFBimportance$Mean_rank), ] #Order by mean rank value.
plot_ly(x=TGFBimportance2$rank_centrality, y=TGFBimportance2$rank_PSMAD2, z=TGFBimportance2$rank_53BP1,
        type="scatter3d", mode="markers+lines",
        marker= list(width = 60, color = TGFBimportance2$Mean_rank, colorscale = "RdBu", opacity=1, sho
        trace=row.names(TGFBimportance2), size=TGFBimportance2$Mean_rank, sizes = c(10, 2000)) %>%
  layout(scene = list(xaxis = list(title = "Centrality rank" ),
                      yaxis = list(title = "pSMAD2 rank"),
                      zaxis = list(title = "53BP1 rank")))
```

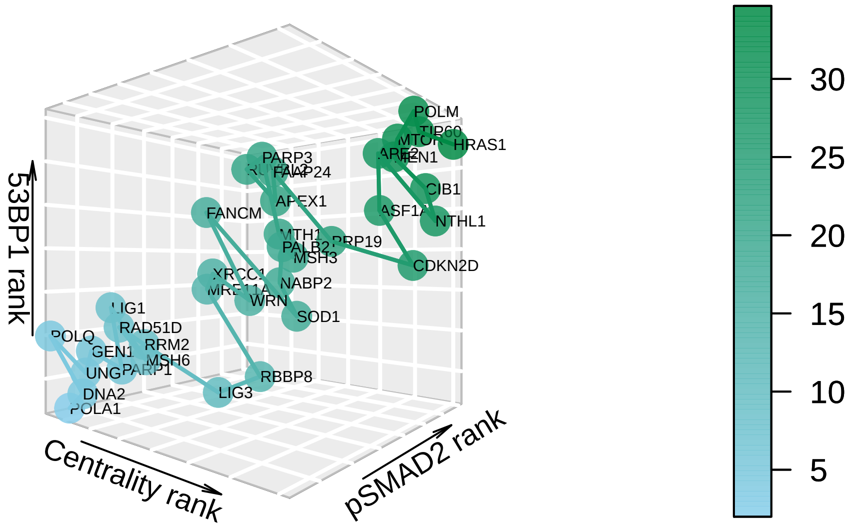
```r
##ALTEJ genes
ALTEJimportance2 <- ALTEJimportance[order(ALTEJimportance$Mean_rank), ] #Order by mean rank value.
plot_ly(x=ALTEJimportance2$rank_centrality, y=ALTEJimportance2$rank_PSMAD2, z=ALTEJimportance2$rank_53B
        type="scatter3d", mode="markers+lines",
        marker= list(width = 60, color = ALTEJimportance2$Mean_rank, colorscale = "RdBu", opacity=1, sh
        trace=row.names(ALTEJimportance2), size=ALTEJimportance2$Mean_rank, sizes = c(10, 2000)) %>%
  layout(scene = list(xaxis = list(title = "Centrality rank" ),
                      yaxis = list(title = "pSMAD2 rank"),
                      zaxis = list(title = "53BP1 rank")))
```

#2. Do a 3D plot of genes' importance with PLOT3D.

```r
##TGFB genes
scatter3D(x=TGFBimportance2$rank_centrality, y=TGFBimportance2$rank_PSMAD2, z=TGFBimportance2$rank_53BP
          type = "b", pch = 20, lwd = 2, cex = 3, #cex = TGFBimportance2$Mean_rank/10
          xlab = "Centrality rank", ylab ="pSMAD2 rank", zlab = "53BP1 rank", cex.lab=0.9,
          colvar=TGFBimportance2$Mean_rank, col =  ramp.col(col = c("skyblue", "hotpink1"), alpha=0.6))
text3D(x=TGFBimportance2$rank_centrality, y=TGFBimportance2$rank_PSMAD2, z=TGFBimportance2$rank_53BP1,
       labels = TGFBimportance2$Gene,
       add = TRUE, colkey = FALSE, cex = 0.5) #Add gene labels.
```



```
##ALTEJ genes
scatter3D(x=ALTEJimportance2$rank_centrality, y=ALTEJimportance2$rank_PSMAD2, z=ALTEJimportance2$rank_53
          type = "b", pch = 20, lwd = 2, cex = 3, #cex = ALTEJimportance2$Mean_rank/7
          xlab = "Centrality rank", ylab ="pSMAD2 rank", zlab = "53BP1 rank", cex.lab=0.9,
          colvar=ALTEJimportance2$Mean_rank, col =  ramp.col(col = c("skyblue", "springgreen4"), alpha=0
text3D(x=ALTEJimportance2$rank_centrality, y=ALTEJimportance2$rank_PSMAD2, z=ALTEJimportance2$rank_53BP1
       labels = ALTEJimportance2$Gene,
       add = TRUE, colkey = FALSE, cex = 0.5) #Add gene labels.
```



PART 12.A: BUBBLECHARTS OF GENES' IMPORTANCE - TGFB GENES

#1. Reorder the genes by their mean weight.

```
TGFBimportance <- arrange(TGFBimportance, TGFBimportance$`mean weight`)
TGFBimportance$Gene <- factor(TGFBimportance$Gene, levels = TGFBimportance$Gene[order(TGFBimportance$`m
table(TGFBimportance$Gene)
```

#2. Create a long formatted table of "Gene, Metric (PSMAD2, 53BP1, centrality, mean weight), value".

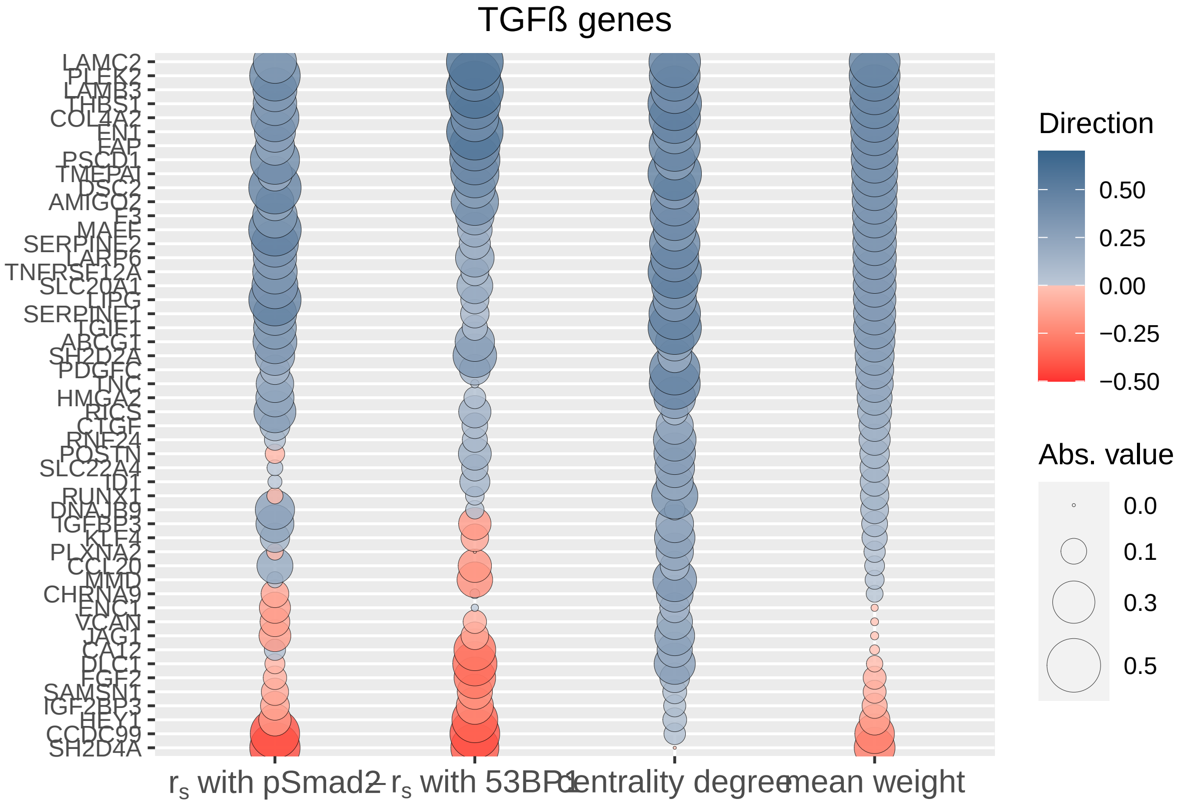
```
A <- TGFBimportance[,c("Gene", "pSmad2", "53BP1", "centrality", "mean weight")]
B <- gather(A, key=Metric, value=value, "pSmad2","53BP1", "centrality", "mean weight", na.rm = FALSE, c
```

#3. Create a variable to indicate the direction of the SCC, turn the SCC into absolute values, and reorder the Metric.

```
B$Direction <- ifelse(B$value<0, "unanticipated", "anticipated")
B$abs_value<- abs(B$value)
B$Metric <- factor(B$Metric, levels = c("pSmad2", "53BP1", "centrality", "mean weight"))
```

#4. Plot a bubblechart of the genes' importance.

```r
ggplot(B, aes(x=Metric, y=Gene, size=abs_value, fill=value)) +
  geom_point(alpha=0.8, shape=21, stroke=0.1) +
  scale_fill_gradientn(colours = c("firebrick1","white","steelblue4"),
                       values = rescale(c(-0.5, 0 , 0.001, 0.7)),
                       guide = "colorbar", limits=c(-0.5, 0.7)) +
  labs(x = "", y = "", fill="Direction", size="Abs. value") +
  ggtitle("TGF  genes") + theme(plot.title = element_text(hjust = 0.5)) +
  theme(axis.text.x = element_text(angle = 0, size = 12)) +
  scale_size_continuous(range = c(0.5,10), breaks = c(0,0.1,0.3, 0.5, 0.6)) +
  scale_x_discrete(name = "", labels = c(expression(r[s]~with~pSmad2),
                                          expression(-r[s]~with~`53BP1`),
                                          "centrality degree", "mean weight")) #PDF 7x8.2.
```



PART 12.B: BUBBLECHARTS OF GENES' IMPORTANCE - ALTEJ GENES

#1. Reorder the genes by their mean weight.

```r
ALTEJimportance <- arrange(ALTEJimportance, ALTEJimportance$`mean weight`)
ALTEJimportance$Gene <- factor(ALTEJimportance$Gene, levels = ALTEJimportance$Gene[order(ALTEJimportanc
table(ALTEJimportance$Gene)
```

#2. Create a long formatted table of "Gene, Metric (PSMAD2, 53BP1, centrality, mean weight), value".

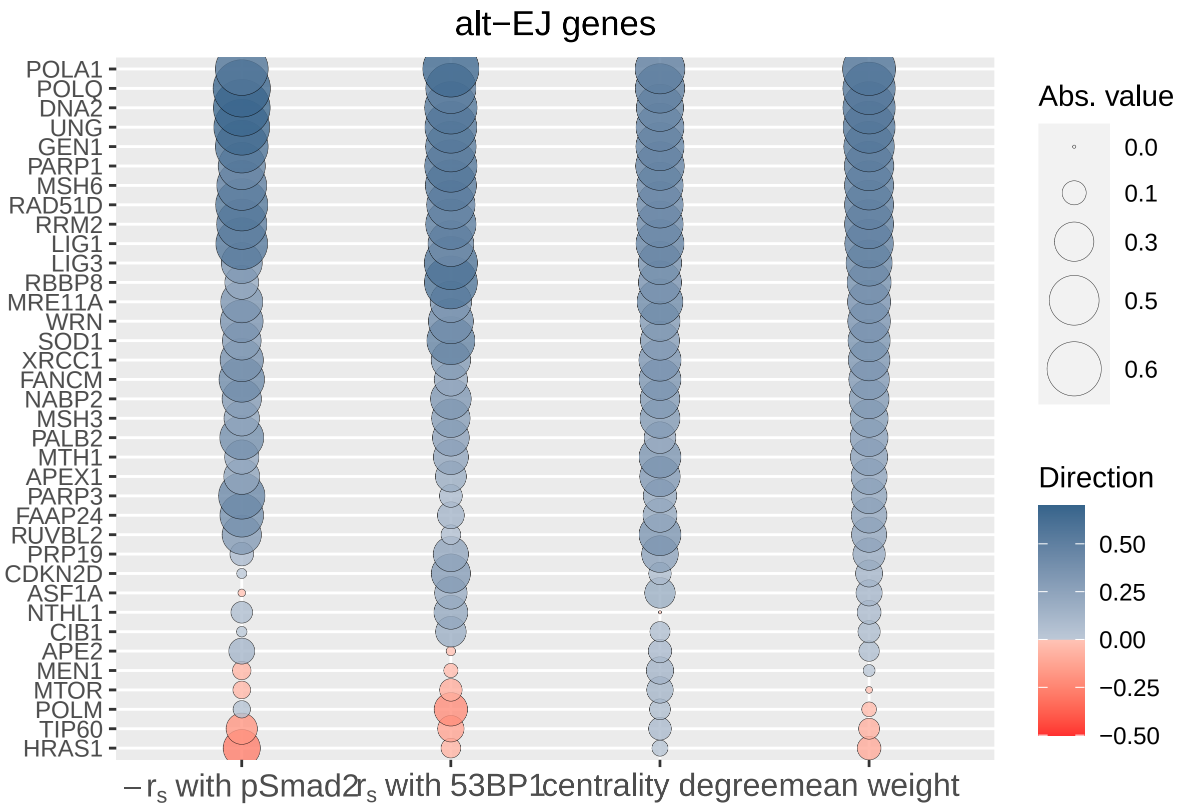
```
A <- ALTEJimportance[,c("Gene", "pSmad2", "53BP1", "centrality", "mean weight")]
B <- gather(A, key=Metric, value=value, "pSmad2","53BP1", "centrality", "mean weight", na.rm = FALSE, c
```

#3. Create a variable to indicate the direction of the SCC, turn the SCC into absolute values, and reorder the Metric.

```
B$Direction <- ifelse(B$value<0, "unanticipated", "anticipated")
B$abs_value<- abs(B$value)
B$Metric <- factor(B$Metric, levels = c("pSmad2", "53BP1", "centrality", "mean weight"))
```

#4. Plot a bubblechart of the genes' importance.

```
ggplot(B, aes(x=Metric, y=Gene, size=abs_value, fill=value)) +
  geom_point(alpha=0.8, shape=21, stroke=0.1) +
  scale_fill_gradientn(colours = c("firebrick1","white","steelblue4"),
                       values = rescale(c(-0.5, 0 , 0.001, 0.7)),
                       guide = "colorbar", limits=c(-0.5, 0.7)) +
  labs(x = "", y = "", fill="Direction", size="Abs. value") +
  ggtitle("alt-EJ genes") + theme(plot.title = element_text(hjust = 0.5)) +
  theme(axis.text.x = element_text(angle = 0, size = 12)) +
  scale_size_continuous(range = c(0.5,10), breaks = c(0,0.1,0.3, 0.5, 0.6)) +
  scale_x_discrete(name = "", labels = c(expression(-r[s]~with~pSmad2),
                                         expression(r[s]~with~`53BP1`),
                                         "centrality degree", "mean weight")) #PDF 7x8.2.
```


alt−EJ genes
POLA1
POLQ
DNA2
UNG
GEN1
PARP1
MSH6
RAD51D
RRM2
LIG1
LIG3
RBBP8
MRE11A
WRN
SOD1
XRCC1
FANCM
NABP2
MSH3
PALB2
MTH1
APEX1
PARP3
FAAP24
RUVBL2
PRP19
CDKN2D
ASF1A
NTHL1
CIB1
APE2
MEN1
MTOR
POLM
TIP60
HRAS1
$-r_s$ with pSmad2
r_s with 53BP1
centrality degree
mean weight
Abs. value
Direction