TADs.R

```
wd <- "/home/mcabrera/Desktop/MN/p53/"
samples <- readRDS(paste0(wd,"samples.rds"))
results <- paste0(wd,"results/HCT116/HiC/")  #path where the clean files are

knitr::opts_chunk$set(dev = "pdf",
                      dpi = 300,
                      # echo = FALSE, #to print code or not
                      cache = TRUE,
                      root.dir = paste0(results))
packages <- c("GenomicRanges", "ggplot2","reshape2", "stringr","dplyr","hrbrthemes","viridis","knitr","
invisible(lapply(packages, library, character.only = TRUE))

colors_samples=c("#fde725ff", "#37b578ff","#21908dff","#31668dff","#43377fff","#440154ff")

# Define function to label summary statistics on the plot
meanFunction <- function(x){
  return(data.frame(y=round(mean(x)), label=round(mean(x,na.rm=T))))}

medianFunction <- function(x){
  return(data.frame(y=round(median(x)), label=round(median(x,na.rm=T))))}
```

```
# Read in data and select relevant columns
TADs_TADbitScore <- read.table(file.path(results, "aligned_TADborders_TADbit_score.tsv"), header = TRUE
# Rename columns
colnames(TADs_TADbitScore) <- c("Chromosome", "TADborder", samples)

# Order data by chromosome and TAD border position and replace NA values with 0
TADs_TADbitScore <- TADs_TADbitScore %>% arrange(Chromosome, TADborder)%>% replace(is.na(.), 0)

#Removing TADborders with no significant border at any timepoint
TADs_TADbitScore <- TADs_TADbitScore[rowSums(TADs_TADbitScore[,-c(1:2)])>0,]

# Remove "chr" from chromosome names
TADs_TADbitScore$Chromosome <- gsub("chr", "", TADs_TADbitScore$Chromosome)

TADs_TADbitScore$individual_TADstate <- data.frame(apply(TADs_TADbitScore[,c(samples)], 2, function(x)
TADs_TADbitScore$Category_tads <- do.call(paste, c(TADs_TADbitScore[,c(length(samples)+3)], sep="-"))

TADs_TADbitScore$state_tads <-as.character(ifelse(TADs_TADbitScore$Category_tads == paste(replicate(leng
```

```
borders_type <- list()
i=0
for (sample in samples){
  while (i < length(samples)-1){
  i=i+1
  print(samples[[i]])
  print(samples[[i+1]])
```

```
  TADs_TADbitScore[[paste0(samples[[i]],"_VS_",samples[[i+1]])]] = ifelse(TADs_TADbitScore$state_tads ==
                                ifelse(TADs_TADbitScore$state_tads == "Variable" & TADs_TADbitScore[[sa
                                ifelse(TADs_TADbitScore$state_tads == "Variable" & TADs_TADbitScore[[sa
                                       ifelse(TADs_TADbitScore$state_tads == "Invariant" & TADs_TADbitS
                                       ifelse(TADs_TADbitScore$state_tads == "Variable" & TADs_TADbitSc

  borders_type[[samples[[i+1]]]] <- table(TADs_TADbitScore[[paste0(samples[[i]],"_VS_",samples[[i+1]])]]
  }
}
```

```
## [1] "Nut.0h"
## [1] "Nut.1h"
## [1] "Nut.1h"
## [1] "Nut.4h"
## [1] "Nut.4h"
## [1] "Nut.7h"
## [1] "Nut.7h"
## [1] "Nut.10h"
## [1] "Nut.10h"
## [1] "Nut.24h"
```

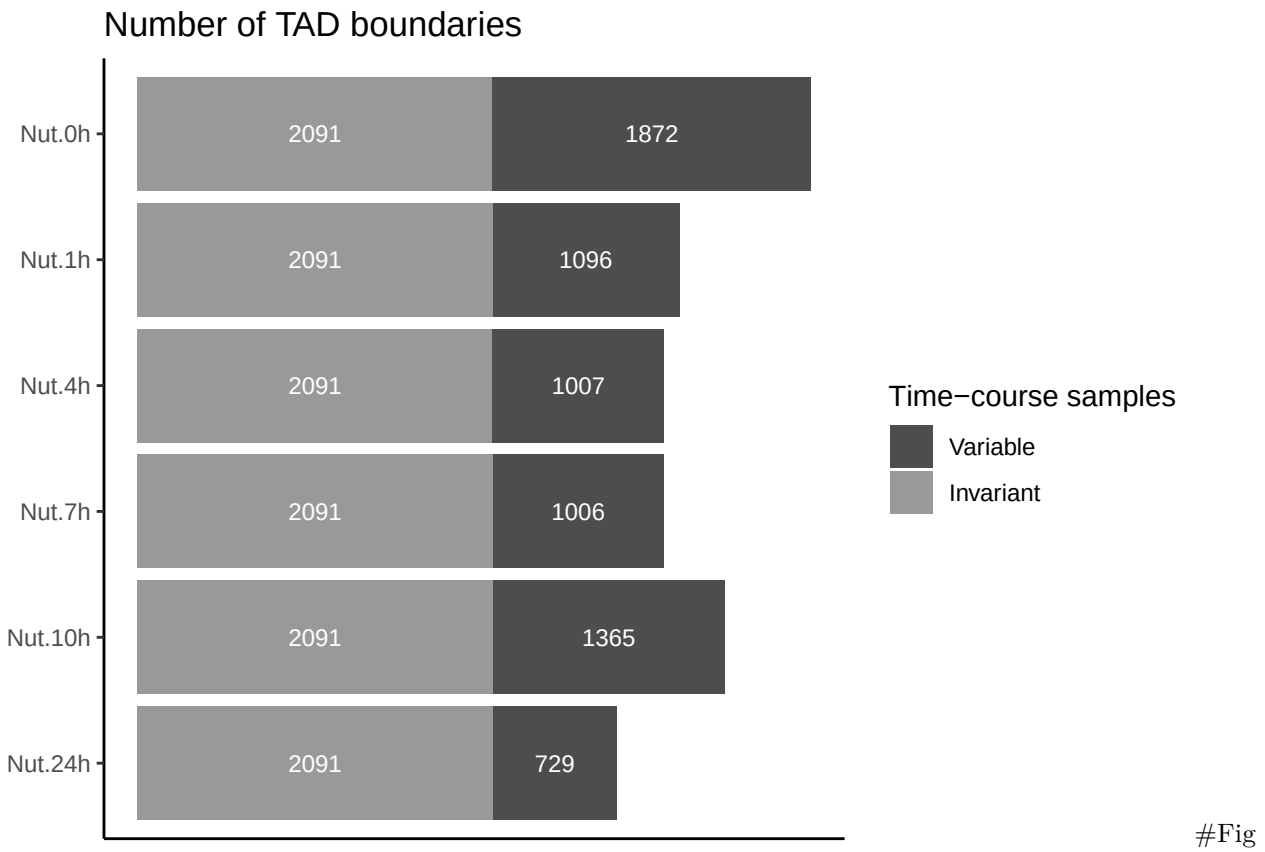
```
borders_type <- data.frame(do.call(rbind,borders_type))
borders_type$samples <- rownames(borders_type)
borders_type$Variable <- borders_type$Gained+borders_type$Maintained

head(borders_type)
```

```
##         Gained Invariant Lost Maintained NA. NoTAD samples Variable
## Nut.1h     234      2091 1010        862 165   415  Nut.1h     1096
## Nut.4h     207      2091  296        800 165  1218  Nut.4h     1007
## Nut.7h     210      2091  211        796 165  1304  Nut.7h     1006
## Nut.10h    621      2091  262        744 165   894 Nut.10h     1365
## Nut.24h    384      2091 1020        345 165   772 Nut.24h      729
```

#Fig 19 A: Bar plot displaying the number of variant and invariant TAD borders along p53 activation. Invariant TAD borders are those detected at all time points. Positive/negative numbers near arrows represent the number of TAD borders gained/lost between the corresponding time points.

```
TADborders_invariant_variable <- borders_type[,grep(pattern="Invariant|Variable|samples", x=colnames(bo
TADborders_invariant_variable_Nut.0h <- data.frame(table(TADs_TADbitScore$individual_TADstate$Nut.0h, TA

TADborders_invariant_variable[paste0(samples[[1]]),] = c(paste0(TADborders_invariant_variable_Nut.0h[TA
                                    paste0(samples[[1]]),
                                   paste0(TADborders_invariant_variable_Nut.0h[TADborders_invariant_varia

TADborders_invariant_variable_m <- melt(TADborders_invariant_variable, id.vars = "samples")
TADborders_invariant_variable_m$variable <- factor(TADborders_invariant_variable_m$variable, levels = c
TADborders_invariant_variable_m$samples <- factor(TADborders_invariant_variable_m$samples, levels = rev

ggplot(TADborders_invariant_variable_m, aes(fill=variable, y=as.numeric(value), x=samples)) +
  geom_bar(position="stack", stat="identity")+
  scale_fill_manual(values=c("grey30","grey60"),name="Time-course samples")+
  ylab("")+
  xlab("")+
  theme_classic()+
```

```
  geom_text(aes(label=value),color = "white",size=3, position = position_stack(vjust = .5))+
  coord_flip()+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank())+
  ggtitle("Number of TAD boundaries")
```



#Fig 19 B: Distribution of TAD sizes along p53 activation.

```
## Loop to calculate the TAD size:
# TADs_TADbitScore <- TADs_TADbitScore %>%
#   mutate(end = NA,
#          id = row_number()) %>%
#   arrange(Chromosome, TADborder)
#
# chr <- TADs_TADbitScore %>%
#   group_split(Chromosome) %>%
#   map(function(x) {
#     x$end[1:(nrow(x) - 1)] <- x$TADborder[2:nrow(x)]
#     x$size <- x$end - x$TADborder
#     return(x)
#   })
#
# chr_split <- bind_rows(chr, .id = "Chromosome")
# c <- list()
# d <- list()
#
# for (sample in samples)
```

```
# {
#   for (chromosome in unique(chr_split$Chromosome))
#   {
#     c[[chromosome]][[sample]] <- chr_split %>% filter(Chromosome == chromosome)
#     c[[chromosome]][[sample]][[paste0("size_",sample)]] <- NA
#     # c[[chromosome]][[sample]][[paste0("location_TAD_",sample)]] <- NA
#
#     # a <- data.frame(c[[chromosome]][[sample]]$TADborder)
#     # a <- a %>% filter(row_number() <= n()-1)
#     # b <- 0
#     # start_TAD <- rbind(b,a)
#     # colnames(start_TAD) <- c("start")
#     # c[[chromosome]][[sample]]$start <- start_TAD$start
#
#     for(row in seq(0, nrow(c[[chromosome]][[sample]])-1, by=1))
#     {
#       row=row+1
#       while (c[[chromosome]][[sample]][[row,sample]] <=4 & row <= nrow(c[[chromosome]][[sample]])-1 )
#           row=row+1
#           print(row)
#         }
#
#       # start_TAD=row
#
#       if (row == nrow(c[[chromosome]][[sample]])-1) {
#         if (c[[chromosome]][[sample]][[row+1,sample]] <=4) {
#           end_border=row
#         } else {
#           end_border=row+1
#         }
#       } else if (row == nrow(c[[chromosome]][[sample]])) {
#         end_border=row
#       } else {
#         end_border=row+1
#       }
#
#       while (c[[chromosome]][[sample]][[end_border,sample]] <=4 & end_border < nrow(c[[chromosome]][[
#           end_border=end_border+1
#           print(end_border)
#         }
#
#       c[[chromosome]][[sample]][[row,paste0("size_",sample)]] = as.numeric(c[[chromosome]][[sample]][[
#
#     }
#   }
# }
#
# # Combine data by chromosome
# distances_chr_split <- bind_rows(c, .id = c("Chromosome"))

# TADs_TADbitScore$start <- distances_chr_split$Nut.0h$start
# TADs_TADbitScore$end <- distances_chr_split$Nut.0h$TADborder
```

```r
# create a data frame with the chromosome column from distances_chr_split
# sizes <- data.frame(Chromosome = distances_chr_split$Chromosome)

# add columns for start, id, and Nut sizes using the mutate function from dplyr
# sizes <- distances_chr_split %>%
#   mutate(TADborder = distances_chr_split$Nut.0h$TADborder,
#          Nut.0h = distances_chr_split$Nut.0h$size_Nut.0h,
#          Nut.1h = distances_chr_split$Nut.1h$size_Nut.1h,
#          Nut.4h = distances_chr_split$Nut.4h$size_Nut.4h,
#          Nut.7h = distances_chr_split$Nut.7h$size_Nut.7h,
#          Nut.10h = distances_chr_split$Nut.10h$size_Nut.10h,
#          Nut.24h = distances_chr_split$Nut.24h$size_Nut.24h)
#
# sizes[is.na(sizes)] <- 0

# write_tsv(sizes, file=paste0(results,"TAD_sizes.tsv"))
sizes <- read_tsv(paste0(results,"TAD_sizes.tsv"),show_col_types = FALSE)

sizes_m <- sizes %>%
  select(Chromosome, start, Nut.0h, Nut.1h, Nut.4h, Nut.7h, Nut.10h, Nut.24h) %>%
  pivot_longer(cols = -c(Chromosome,start), names_to = "Timepoint", values_to = "Size") %>%
  filter(Size < 5*10^6 & Size > 0)

sizes_m$Timepoint <- factor(sizes_m$Timepoint, levels = c("Nut.0h", "Nut.1h","Nut.4h", "Nut.7h", "Nut.1

my_comparisons <- list(c("Nut.0h","Nut.1h"),c("Nut.0h","Nut.10h"),c("Nut.1h","Nut.10h"))

# Plot TAD sizes by timepoint
Fig2B <- sizes_m %>%
  ggplot(aes(x = Timepoint, y = Size/1000, color = Timepoint, fill=Timepoint)) +
  geom_violin(width = 0.9, fill="white") +
  geom_boxplot(width = 0.2, color="black") +
  scale_color_manual(values = colors_samples) +
  scale_fill_manual(values = colors_samples) +
  theme_ipsum() +
  theme_classic() +
  theme(legend.position = "none", plot.title = element_text(size = 11)) +
  xlab("") +
  ylab("TAD size in kb") +
  ylim(c(0, 2000)) +
  # stat_compare_means(method = "wilcox.test", comparisons = list(c("Nut.0h","Nut.1h"),c("Nut.0h","Nut.
  stat_compare_means(label = "p.signif", method = "wilcox.test", ref.group = "Nut.0h",legend='right') +
  # stat_compare_means(comparisons = statistical_comparision_compartments,method = "wilcox.test",label
  # stat_summary(fun.data = meanFunction, geom = "text", color = "black", size = 2.5, vjust = 1.3)
    stat_summary(fun.data = medianFunction, geom = "text", color = "black", size = 2.5, vjust = 1.3)
```

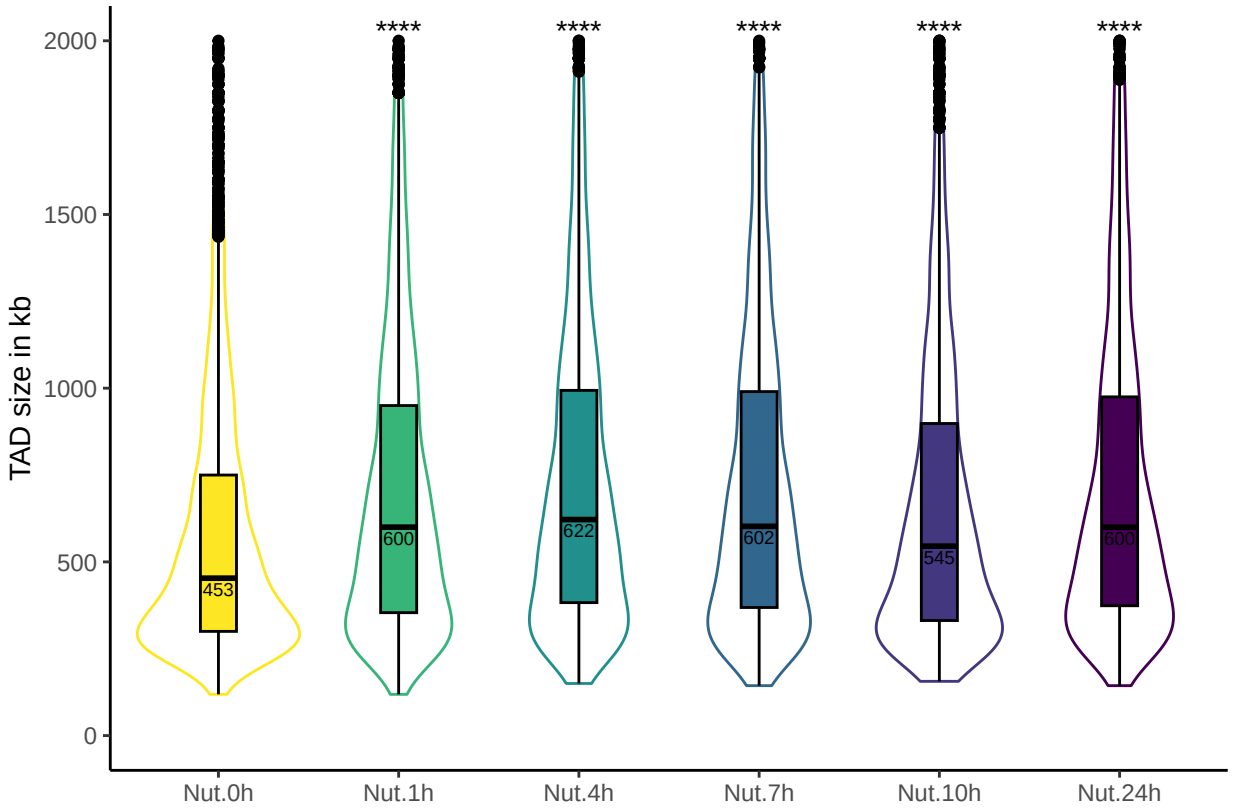
```
## Warning in stat_compare_means(label = "p.signif", method = "wilcox.test", :
## Ignoring unknown parameters: `legend`
```

```r
# stat_compare_means(comparisons = my_comparisons)
print(Fig2B)
```

```
## Warning: Removed 1268 rows containing non-finite values (`stat_ydensity()`).
## Warning: Removed 1268 rows containing non-finite values (`stat_boxplot()`).
```

```
## Warning: Removed 1268 rows containing non-finite values
## (`stat_compare_means()`).
```

```
## Warning: Removed 1268 rows containing non-finite values (`stat_summary()`).
```




```
# Load the TAD insulation score data
TADs_InsulationScore <- read.table(paste0(results,"aligned_TADborders_InsulationScore_TADbit.tsv"),sep =
colnames(TADs_InsulationScore) <- c("rowid","Chromosome", "Start","End","Nut.0h","Nut.1h","Nut.4h","Nut

# Remove the rowid column and replace -Inf values with NA
TADs_InsulationScore <- TADs_InsulationScore[, -1]
TADs_InsulationScore[TADs_InsulationScore == -Inf] <- NA

TADs_InsulationScore$Chromosome <- gsub("chr", "", TADs_InsulationScore$Chromosome)

TADs_sigmoidInsulationScore <- sigmoid(TADs_InsulationScore[, !(colnames(TADs_InsulationScore) %in% c("
TADs_sigmoidInsulationScore$Chromosome <- TADs_InsulationScore$Chromosome
TADs_sigmoidInsulationScore$Start <- TADs_InsulationScore$Start
TADs_sigmoidInsulationScore$End <- TADs_InsulationScore$End

# # Adding the insulation score values to the TADbit score to have a table with all the information: TA
# IS <- list()
# df <- list()
#
# library(tictoc)
#
# # Define the number of iterations to be performed
# n_iter <- length(samples) * nrow(TADs_TADbitScore)
#
```

```
# # Start the timer
# tic()
#
# # Initialize the progress bar
# pb <- txtProgressBar(min = 0, max = n_iter, style = 3)
#
# for (sample in samples)
#   {
#   # print(sample)
#   for (i in seq(1:nrow(TADs_TADbitScore)))
#     {
#     # Update the progress bar
#     setTxtProgressBar(pb, (i - 1) * length(samples) + match(sample, samples))
#
#     # print(i)
#     if (TADs_TADbitScore[i,][[sample]] > 4)
#       {
#       # print("tad border > 4")
#       if (any(TADs_InsulationScore$Chromosome == TADs_TADbitScore[i,]$Chromosome & TADs_InsulationSco
#         {
#         # print("tad border exacto")
#         IS[[paste0("IS_",sample)]][[i]] <- round(as.numeric(TADs_InsulationScore[c(TADs_InsulationSco
#       } else {
#         # print("tad border no exacto")
#         IS[[paste0("IS_",sample)]][[i]] <- round(as.numeric(TADs_InsulationScore[c(TADs_InsulationSco
#       }
#     } else {
#       # IS[[paste0("IS_",sample)]][[i]] <- as.numeric(0)
#       IS[[paste0("IS_",sample)]][[i]] <- NA
#     }
#   }
#   df[[sample]] <- do.call("rbind",IS[[paste0("IS_",sample)]])
# }
#
# # Stop the timer and print the elapsed time
# toc()
#
# TADborder_IS <- data.frame(do.call("cbind",IS))
# TADborder_IS <- as.data.frame(apply(TADborder_IS, 2, as.numeric))
# TADborder_IS$TADborder <- TADs_TADbitScore$TADborder
# TADborder_IS$Chromosome <- TADs_TADbitScore$Chromosome
#
# # Apply sigmoid transformation to the insulation scores
# TADborder_sigIS <- sigmoid(TADborder_IS[, !(colnames(TADborder_IS) %in% c("TADborder", "Chromosome"))]
# colnames(TADborder_sigIS) <- paste("sigmoid_",colnames(TADborder_sigIS), sep = "")
# TADborder_sigIS$TADborder <- TADs_TADbitScore$TADborder
# TADborder_sigIS$Chromosome <- TADs_TADbitScore$Chromosome
#
# # merging IS y TADbit score con más info de los TADs
# TADbitScore_IS_ISsig <- merge(TADborder_IS,TADborder_sigIS, by=c("Chromosome","TADborder"))
# TADs_TADbitScore_IS_ISsig <- merge(TADs_TADbitScore,TADbitScore_IS_ISsig, by=c("Chromosome","TADborde
#
# write_tsv(TADs_TADbitScore_IS_ISsig[,-9], file = paste0(results,"aligned_TADborders_TADbitScore_IS_IS
```

```
TADs_TADbitScore_IS_ISsig <- read_tsv(file = paste0(results,"aligned_TADborders_TADbitScore_IS_IS_sig.ts

# Dividing Variant and Invariant TADs
TADs_TADbitScore_IS_ISsig_Invariant <- data.frame(TADs_TADbitScore_IS_ISsig[grep("Invariant", TADs_TADbi
TADs_TADbitScore_IS_ISsig_Variant <- data.frame(TADs_TADbitScore_IS_ISsig[grep("Variable", TADs_TADbitSc

TADs_TADbitScore_IS_ISsig_Variant$types <-as.character(

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == paste(replicate(length(samples), "NoTAD")

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-NoTAD-NoTAD-NoTAD-NoTAD-NoTAD", 'Nut

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-TAD-TAD-TAD-TAD-TAD", 'p53 activati

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-NoTAD-TAD-TAD-TAD-TAD", 'Nut.1h speci

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-TAD-NoTAD-NoTAD-NoTAD-NoTAD", 'Nut

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-TAD-NoTAD-TAD-TAD-TAD", 'Nut.4h speci

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-NoTAD-TAD-NoTAD-NoTAD-NoTAD", 'Nut

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-TAD-TAD-NoTAD-TAD-TAD", 'Nut.7h speci

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-NoTAD-NoTAD-TAD-NoTAD-NoTAD", 'Nut

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-TAD-TAD-TAD-NoTAD-TAD", 'Nut.10h spe

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-NoTAD-NoTAD-NoTAD-TAD-NoTAD", 'Nut

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-TAD-TAD-TAD-TAD-NoTAD", 'Nut.24h spe

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-NoTAD-NoTAD-NoTAD-NoTAD-TAD", 'Nut

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-TAD-TAD-TAD-NoTAD-NoTAD" | TADs_TA

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-NoTAD-NoTAD-NoTAD-TAD-TAD" | TADs_TA

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "NoTAD-NoTAD-NoTAD-NoTAD-TAD-TAD", 'Late a

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-TAD-TAD-TAD-NoTAD-NoTAD", 'Late disa

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-NoTAD-NoTAD-NoTAD-TAD-NoTAD", 'Nut.0

    ifelse(TADs_TADbitScore_IS_ISsig_Variant$Category_tads == "TAD-TAD-NoTAD-NoTAD-NoTAD-NoTAD", 'Nut.0

           'Highly dynamic')))))))))))))))))))))

TADs_TADbitScore_IS_ISsig_Variant_freq <- data.frame(sort(table(TADs_TADbitScore_IS_ISsig_Variant$types
TADs_TADbitScore_IS_ISsig_Variant_freq$type_freq <- TADs_TADbitScore_IS_ISsig_Variant_freq$Freq*100/nrow
TADs_TADbitScore_IS_ISsig_Variant_freq$name_freq <- as.character(TADs_TADbitScore_IS_ISsig_Variant_freq$
TADs_TADbitScore_IS_ISsig_Variant_freq$name_freq[TADs_TADbitScore_IS_ISsig_Variant_freq$type_freq <5] <-
```

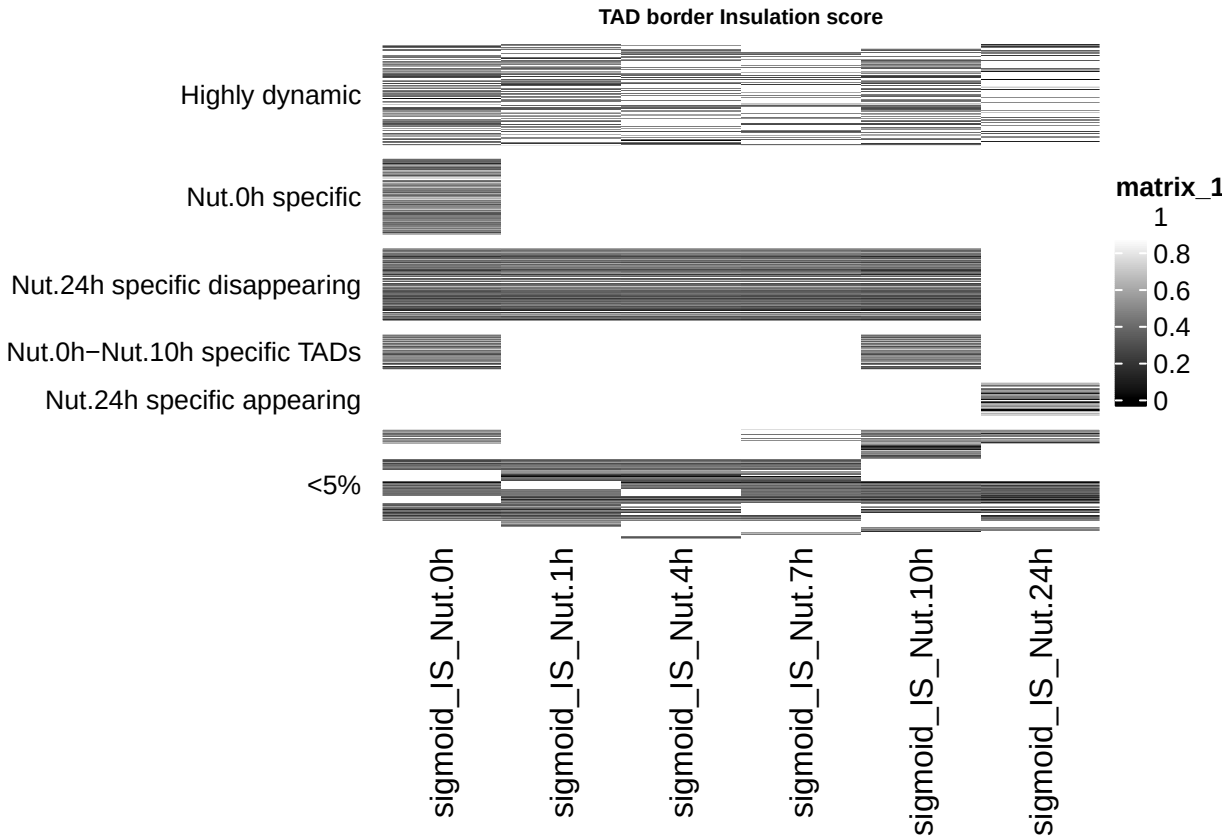
```
TADs_TADbitScore_IS_ISsig_Variant_percentage_sum <- data.frame(tapply(TADs_TADbitScore_IS_ISsig_Variant
TADs_TADbitScore_IS_ISsig_Variant_percentage_sum$samples <- rownames(TADs_TADbitScore_IS_ISsig_Variant_
colnames(TADs_TADbitScore_IS_ISsig_Variant_percentage_sum) <- c("Percentage", "Samples")
TADs_TADbitScore_IS_ISsig_Variant_percentage_sum <- TADs_TADbitScore_IS_ISsig_Variant_percentage_sum[or

TADs_TADbitScore_IS_ISsig_Variant$heatmap = TADs_TADbitScore_IS_ISsig_Variant_freq$type_freq[match(TADs

TADs_TADbitScore_IS_ISsig_Variant$heatmap_types = TADs_TADbitScore_IS_ISsig_Variant_freq$name_freq[matc

TADs_TADbitScore_IS_ISsig_Variant <- TADs_TADbitScore_IS_ISsig_Variant[order(TADs_TADbitScore_IS_ISsig_

TADs_TADbitScore_IS_ISsig_Variant$Rownames <- paste0(TADs_TADbitScore_IS_ISsig_Variant$types,"\n",round
```

```
# a <- list()
# IS <- list()
# b <- list()
#
# InsulationScore <- TADs_sigmoidInsulationScore
#
# pb <- progress_bar$new(
#     format = "[:bar] :percent :current/:total | :elapsed eta: :eta",
#     total = length(samples) * length(seq(-500000, 500000, by = 50000))
#   )
#
# for (sample in samples)
# {
#   # print(sample)
#    pb$tick() # update progress bar
#
#     for (distance in seq(-500000,500000,by=50000))
#     {
#      # print(distance)
#      d=distance/1000
#      options(scipen=999) # this is used to remove scientific notation in printing
#
#       for (i in seq(1:nrow(TADs_TADbitScore)-1))
#       {
#         # print(i)
#         if (TADs_TADbitScore[i,][[sample]] > 4)
#         {
#           # print("tad >4")
#           if (any(InsulationScore$Chromosome == TADs_TADbitScore[i,]$Chromosome & InsulationScore$Sta
#             # print("tadborder exacto")
#             IS[[sample]][[paste0(sample,"_",d,"kb")]][[i]] <- as.numeric(InsulationScore[c(Insulation
#           else{
#             # print("tadborder no exacto")
#             IS[[sample]][[paste0(sample,"_",d,"kb")]][[i]] <- as.numeric(InsulationScore[c(Insulation
#         }else{
#             # print("tad <4")
#             IS[[sample]][[paste0(sample,"_",d,"kb")]][[i]] <- NA
#           }
#       }
#      }
#     a[[sample]] <- do.call("cbind",IS[[sample]])
```

```
#     rownames(a[[sample]]) <- TADs_TADbitScore$TADborder
# }
#
# average_IS_TADs <- data.frame(do.call("cbind",a),check.names = FALSE)
# average_IS_TADs <- data.frame(apply(average_IS_TADs,2,as.numeric),check.names = FALSE)
# average_IS_TADs$TADborder <- TADs_TADbitScore$TADborder
# average_IS_TADs$state_tads <- TADs_TADbitScore$state_tads
#
# write_tsv(average_IS_TADs, file = paste0(results,"average_sigmoidIS_All_TADs.tsv"))
average_IS_TADs <- read_tsv(paste0(results,"average_sigmoidIS_All_TADs.tsv"),show_col_types = FALSE)

# average_IS_TADs_median <- data.frame(apply(average_IS_TADs[, !names(average_IS_TADs) %in% c("TADborde
# foo <- data.frame(str_split_fixed(rownames(average_IS_TADs_median), "_", 2))
# average_IS_TADs_median$samples <- foo$X1
# average_IS_TADs_median$distances <- as.factor(foo$X2)
# colnames(average_IS_TADs_median) <- c("values", "samples", "distances")

# write_tsv(average_IS_TADs_median, file = paste0(results,"Median_average_sigmoidIS_All_TADs.tsv"))

average_IS_TADs_median <- read_tsv(paste0(results,"Median_average_sigmoidIS_All_TADs.tsv"),show_col_typ
```

#Fig 19C: Clustering of TAD borders based on their TAD insulation scores and the patterns of temporal changes observed throughout p53 activation. Only TAD borders characterized by a TADbit score > 4 were included. TAD borders were manually clustered considering their dynamism throughout p53 activation.

```
Heatmap(as.matrix(TADs_TADbitScore_IS_ISsig_Variant[,grep(pattern="sigmoid_IS_", x=colnames(TADs_TADbit
row_title_rot = 0,row_title_gp = gpar(col = c("black"),fontsize=10),row_gap = unit(1.5, "mm")
,border = TRUE,border_gp = gpar(col = "white", lwd = 1),
cluster_columns=F,col = c("black","white"),na_col = "white")
```



Extra # {r V plot all TADborders} # average_IS_TADs_median$distances <- factor(average_IS_TADs_median$dis
levels=c("-500kb","-450kb","-400kb","-350kb","-300kb","-250kb","-200kb","-150kb","-100kb","-50kb","0kb"
# # average_IS_TADs_median$samples <- factor(average_IS_TADs_median$samples, levels =
c("Nut.0h", "Nut.1h", "Nut.4h", "Nut.7h", "Nut.10h", "Nut.24h")) # # # Basic line plot
with points # ggplot(data=average_IS_TADs_median, aes(x=distances, y=values, group=samples,
color=samples)) + # geom_line(aes(col=samples))+ # scale_x_discrete(guide = guide_axis(angle
= 90)) + # scale_color_manual(values=c(colors_samples))+ # theme_classic()+ # ggtitle("All
TAD borders")+ # ylim(c(0.3,0.6)) # # # {r V plot invariant TADborders} # average_ISsig_TADs_Invariant
<- average_IS_TADs[average_IS_TADs$state_tads == "Invariant",] # # average_ISsig_TADs_Invariant_median
<- data.frame(apply(average_ISsig_TADs_Invariant[, !names(average_IS_TADs) %in% c("TADborder","state_ta
2, function(x) {median(x,na.rm = TRUE)}),check.names = FALSE) # foo <- data.frame(str_split_fixed(rownar
"_", 2)) # average_ISsig_TADs_Invariant_median$samples <- foo$X1 # average_ISsig_TADs_Invariant_median$
<- foo$X2 # colnames(average_ISsig_TADs_Invariant_median) <- c("values", "samples",
"distances") # # average_ISsig_TADs_Invariant_median$distances <- factor(average_ISsig_TADs_Invariant_
levels=c("-500kb","-450kb","-400kb","-350kb","-300kb","-250kb","-200kb","-150kb","-100kb","-50kb","0kb"
# # average_ISsig_TADs_Invariant_median$samples <- factor(average_ISsig_TADs_Invariant_median$samples,
levels = c("Nut.0h", "Nut.1h", "Nut.4h", "Nut.7h", "Nut.10h", "Nut.24h")) # # # Basic
line plot with points # ggplot(data=average_ISsig_TADs_Invariant_median, aes(x=distances,
y=values, group=samples, color=samples)) + # geom_line(aes(col=samples))+ # scale_x_discrete(guide
= guide_axis(angle = 90)) + # scale_color_manual(values=c(colors_samples))+ # theme_classic()+
ggtitle("Invariant TAD borders")+ # ylim(c(0,1)) # # # {r V plot Variant TAD
borders} # average_ISsig_TADs_variable <- average_IS_TADs[average_IS_TADs$state_tads ==
"Variable",] # # average_ISsig_TADs_variable_median <- data.frame(apply(average_ISsig_TADs_variable[,
!names(average_IS_TADs) %in% c("TADborder","state_tads")], 2, function(x) {median(x,na.rm
= TRUE)}),check.names = FALSE) # foo <- data.frame(str_split_fixed(rownames(average_ISsig_TADs_variable_
"_", 2)) # average_ISsig_TADs_variable_median$samples <- foo$X1 # average_ISsig_TADs_variable_median$dis
<- foo$X2 # colnames(average_ISsig_TADs_variable_median) <- c("values", "samples", "distances")
average_ISsig_TADs_variable_median$distances <- factor(average_ISsig_TADs_variable_median$distances

```
levels=c("-500kb","-450kb","-400kb","-350kb","-300kb","-250kb","-200kb","-150kb","-100kb","-50kb","0kb"
#  # average_ISsig_TADs_variable_median$samples <- factor(average_ISsig_TADs_variable_median$samples,
levels = c("Nut.0h", "Nut.1h", "Nut.4h", "Nut.7h", "Nut.10h", "Nut.24h")) #  # # Basic
line plot with points # ggplot(data=average_ISsig_TADs_variable_median, aes(x=distances,
y=values, group=samples, color=samples)) + #   geom_line(aes(col=samples))+ #   scale_x_discrete(guide
= guide_axis(angle = 90)) + #   scale_color_manual(values=c(colors_samples))+ #   theme_classic()+
#   ggtitle("Variant TAD borders")+ #   ylim(c(0,1)) #
```