Code for generating the data underlying the scRNA-seq-based figures in the SIFT-seq paper

Friederike Dündar

2/7/2022; updated 2022-02-09

Contents

```r
library(SingleCellExperiment); library(data.table);library(magrittr)
library(ggplot2); theme_set(theme_bw(base_size = 16) )
library(ggrepel); library(patchwork)
```

For each patient's data set (0606T1, 21LT2, MR4050/MK_02), there's a separate R package that contains the processed data as well as the code details of how that data was generated (check out the vignettes for that).

Here, we show how the figures shown in the SIFT-seq manuscript were generated although the final-final versions that made it into the paper were also done with GraphPadPrism for reasons of legibility.

```r
## load winners
data("cdrs0606T1", package = "Klebanoff0606T1")
data("cdrs21LT2", package = "Klebanoff21LT2")

## load SingleCellExperiment objects
sce.06 <- Klebanoff0606T1::load_0606T1shared()
Klebanoff0606T1::load_DE_results() #delist.both
de.06 <- delist.both

sce.21 <- Klebanoff21LT2::load_21LT2shared()
Klebanoff21LT2::load_DE_results() #delist.both
de.21 <- delist.both

rm(delist.both); invisible(gc())
```

Fig 1

- (B) and (E): "tgrams" of IFNG logFC for 21LT2 and 0606T1, respectively
- (C) and (D): volcano plots and violin plots were done in GraphPad Prism, see the xlsx and txt files in the data directory

```r
t21  <- Klebanoff0606T1::prep_data4tgram(sce.21,
  which_clonotypes = unique(sce.21$id),
  DEresult_list = de.21, goi = "IFNG",
  additional_colData = c("antigen","freq_per_Sample", "freq_across_all",
    "fit_for_test", "Patient"))

t06 <- Klebanoff0606T1::prep_data4tgram(sce.06,
  which_clonotypes = unique(sce.06$id),
  DEresult_list = de.06, goi = "IFNG",
  additional_colData = c("antigen","freq_per_Sample", "freq_across_all",
    "fit_for_test", "Patient"))
```

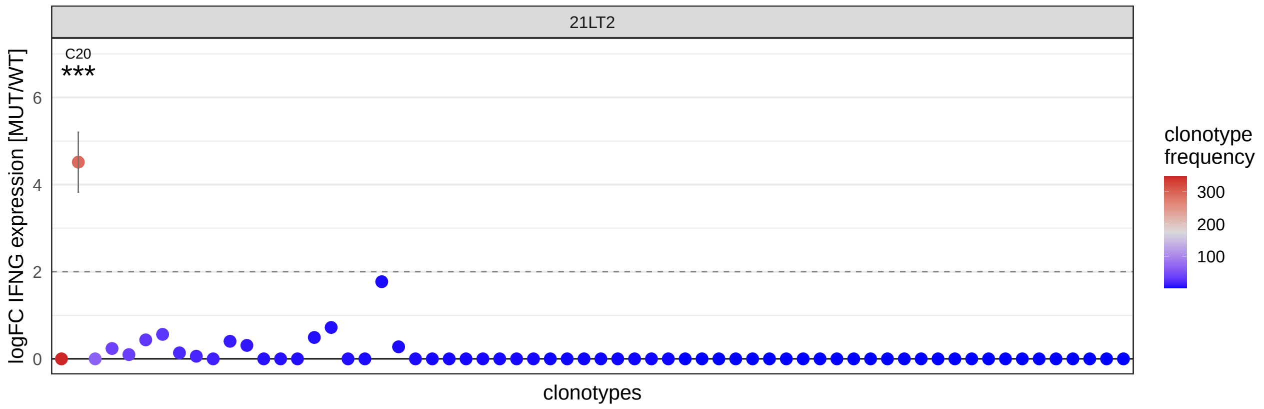
```
#' Extract T test results
#' @description returns the relevant values calculated
#' by R's \code{\link{t.test()}} in the form of a data.table.
#' @return data.table with logFC value (MUT/WT), standard error,
#' lower and upper bounds of the confidence interval.
extract.tres <- function(testres){
  logFC.t = testres$estimate[2] - testres$estimate[1] #levels: WT < MUT
  sde = testres$stderr
  ci.low = testres$conf.int[[2]]*-1 # needed b/c WT < MUT levels
  ci.up = testres$conf.int[[1]]*-1
  return(data.table(logFC.t = logFC.t, sde = sde, ci.low = ci.low, ci.up = ci.up))
}
```

```
# 21LT2 ===================================
## calculate CI
test21res <- t21[
  fit_for_test == TRUE ,
  extract.tres(t.test(logcounts~antigen)),
  by = c("id", "gene_name")]

## summarize everything in one dt
t21.summ <- t21[, -c("cell", "cdr3s_aa","logcounts", "Sample","antigen", "freq_per_Sample"),
  with=FALSE] %>%
  unique %>% test21res[., on = "id"] %>% unique
t21.summ[ is.na(ci.low), ci.low:=0]
t21.summ[ is.na(ci.up), ci.up:=0]


# 0606T1 ===================================
## calculate CI etc.
test06res <- t06[
  fit_for_test == TRUE ,
  extract.tres(t.test(logcounts~antigen)),
  by = c("id", "gene_name")]

## summarize
t06.summ <- t06[, -c("cell", "cdr3s_aa","logcounts", "Sample","antigen", "freq_per_Sample"),
  with=FALSE] %>%
  unique %>% test06res[., on = "id"] %>% unique
t06.summ[ is.na(ci.low), ci.low:=0]
t06.summ[ is.na(ci.up), ci.up:=0]

t21.summ0 <- copy(t21.summ)
t21.summ0[, logFC := ifelse(logFC.t > 0, logFC.t, 0)]
```

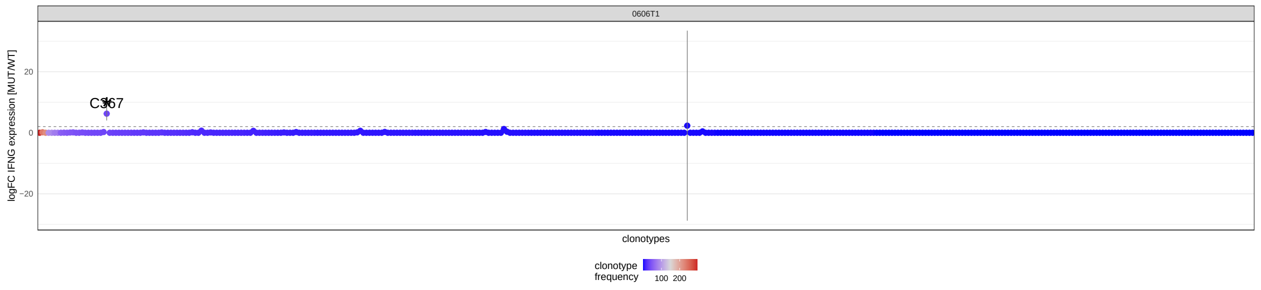
```r
t21.summ0[, logFC := ifelse(is.na(logFC.t), 0, logFC)]
t21.summ0[, ci.low0 := ifelse(logFC.t >0, ci.low, 0)]
t21.summ0[, ci.up0 := ifelse(logFC.t >0, ci.up, 0)]
setorder(t21.summ0, -freq_across_all)
t21.summ0$id.sort <- factor(t21.summ0$id,
  levels = t21.summ0$id, ordered = TRUE)

t06.summ0 <- copy(t06.summ)
t06.summ0[, logFC := ifelse(logFC.t > 0, logFC.t, 0)]
t06.summ0[, logFC := ifelse(is.na(logFC.t), 0, logFC)]
t06.summ0[, ci.low0 := ifelse(logFC.t >0, ci.low, 0)]
t06.summ0[, ci.up0 := ifelse(logFC.t >0, ci.up, 0)]
setorder(t06.summ0, -freq_across_all)
t06.summ0$id.sort <- factor(t06.summ0$id, levels = t06.summ0$id, ordered = TRUE)
```

```r
thresh <- 2
```

```r
ystar <- max(t21.summ0$logFC) + 2

ggplot(t21.summ0, aes(x = id.sort, y = logFC)) +
  geom_hline(yintercept = 0) +
  geom_hline(yintercept = thresh, linetype = "dashed", color = "grey50") +
  ## points ------------------------------
  geom_point(size = 4, aes(color = freq_across_all)) +
  scale_color_gradientn(
    name = "clonotype\nfrequency",
    colours = c("blue","grey85","firebrick3")) +
  ## error bars ---------------------------
  geom_errorbar(data = t21.summ0[logFC >= thresh],
    aes(ymin=ci.low0, ymax=ci.up0),
    width=.1, color="grey45") +
  facet_grid(. ~ Patient, scales="free_x", space = "free_x") +
  ## add asterisks -----------------------
  geom_text(inherit.aes = FALSE,
    aes(x = id.sort, y = ystar, label=star),
    colour="black", size=8)+
  geom_text(data = t21.summ0[star != ""],
    inherit.aes = FALSE,
    aes(x = id.sort, label = id, y = ystar+.5)) +
  ## grid appearance etc. ---------------
  theme(panel.grid.minor.x=element_blank(),
    panel.grid.major.x=element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_blank()) +
  xlab("clonotypes") +
  ylab("logFC IFNG expression [MUT/WT]")
```



```
ystar <- max(t06.summ0$logFC) + 2.5 ## DF

ggplot(t06.summ0, aes(x = id.sort, y = logFC)) + ## DF
  geom_hline(yintercept = 0) +
  geom_hline(yintercept = thresh, linetype = "dashed", color = "grey50") +
  ## points --------------------------------
  geom_point(size = 4, aes(color = freq_across_all)) +
  scale_color_gradientn(
    name = "clonotype\nfrequency",
    colours = c("blue","grey85","firebrick3")) +
  ## error bars ---------------------------
  geom_errorbar(data = t06.summ0[logFC >= thresh], ## DF
    aes(ymin=ci.low0, ymax=ci.up0),
    width=.1, color="grey45") +
  facet_grid(. ~ Patient, scales="free_x", space = "free_x") +
  ## add asterisks -----------------------
  geom_text(inherit.aes = FALSE,
    aes(x = id.sort, y = ystar, label=star),
    colour="black", size=15)+
  geom_text(data = t06.summ0[star != ""], ## DF
    inherit.aes = FALSE, size = 8,
    aes(x = id.sort, label = id, y = ystar+1)) +
  ## grid appearance etc. ---------------
  theme(panel.grid.minor.x=element_blank(),
    panel.grid.major.x=element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_blank(),
    legend.position = "bottom") +
  xlab("clonotypes") +
  ylab("logFC IFNG expression [MUT/WT]")
```



Ext Fig 1: volcano plots MUT vs. WT

–> see the data files `DEgenes_21LT2_C18_mut_vs_wt.txt` and `DEgenes_0606T1.xlsx`.

Ext Fig 3 and 4: boxplots of IFNg

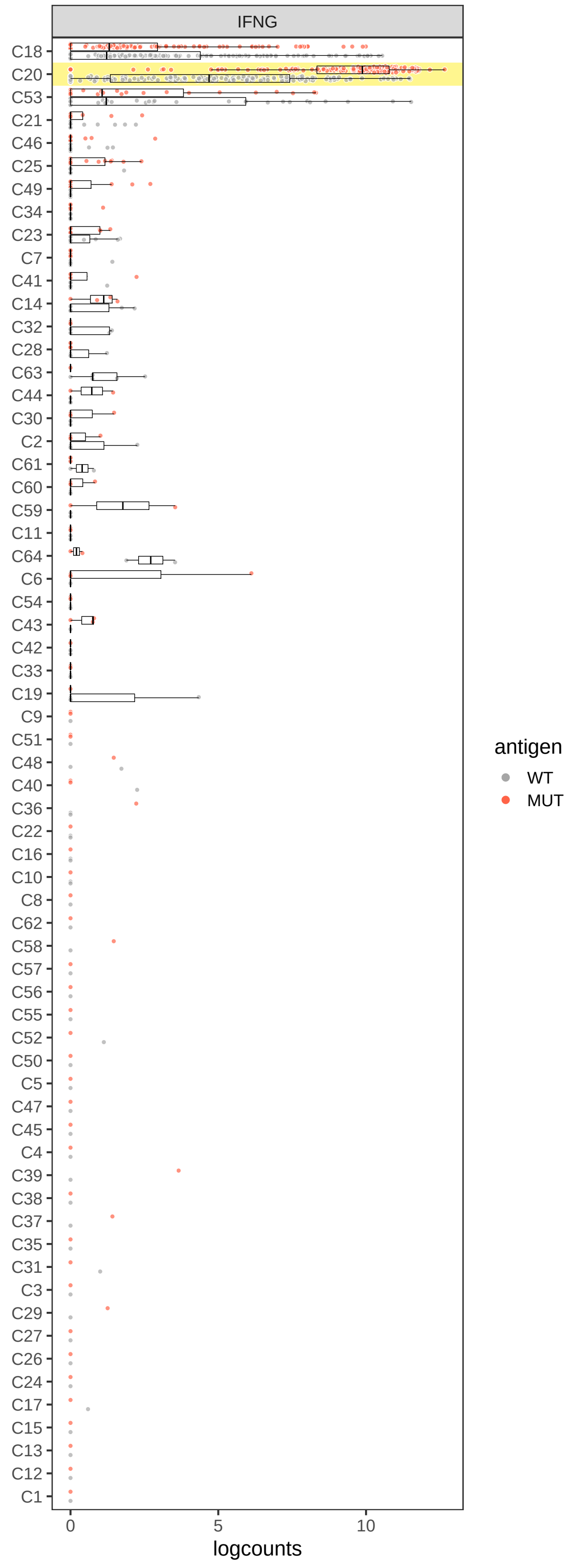
> mutation-reactive TCRs can be identified by normalized IFNG signal comparisons between MUT and WT stimulated conditions

- ExtFig 3: 21LT1
- ExtFig 4: 0606T1:

```
t21  <- Klebanoff0606T1::prep_data4tgram(sce.21,
  which_clonotypes = unique(sce.21$id),
  DEresult_list = de.21, goi = "IFNG",
  additional_colData = c("antigen","freq_per_Sample", "freq_across_all",
    "fit_for_test", "Patient"))
setorder(t21, freq_across_all)
t21$id <- factor(t21$id, levels = unique(t21$id), ordered = TRUE)

ggplot(t21, aes(x = id, y = logcounts))+
  geom_point(alpha = 0, aes(color = antigen)) +
  geom_tile(data=t21[star != ""],
      aes(x = id, y = 1, height = Inf, width = 1),
      alpha = 1, fill = "khaki1") +
  ggbeeswarm::geom_quasirandom(
   groupOnX = TRUE, size = 1.5, shape = 21, alpha =.7,
    dodge.width = .8, aes(fill = antigen), color = "white") +
  geom_boxplot(data=t21[freq_across_all > 3 ],
    outlier.alpha = 0,lwd=.25, fill = NA, aes(color = antigen)) +
  facet_grid(.~gene_name) + coord_flip() +
  xlab("") +
  theme(panel.grid = element_blank()) +
  scale_fill_manual(values = c("grey65","tomato1")) +
  scale_color_manual(values = c("black","black")) +
  ggtitle("All clonotypes of 21LT2",
    subtitle = "sorted by frequency; yellow highlights the logFC\nthat
    were found to be statistically significant") +
   guides(fill = guide_legend(override.aes = list(size = 3, alpha = 1) ),
      color = FALSE )
```

All clonotypes of 21LT2

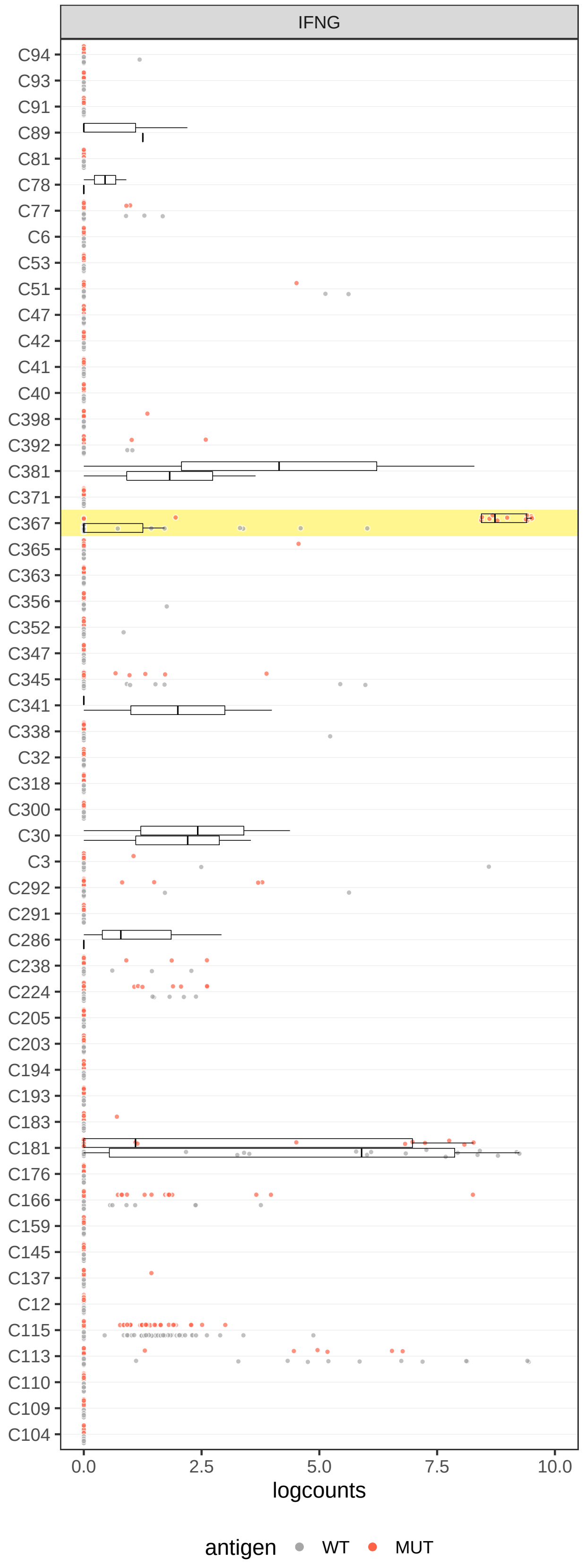
sorted by frequency; yellow highlights the logFC
that
    were found to be statistically significant



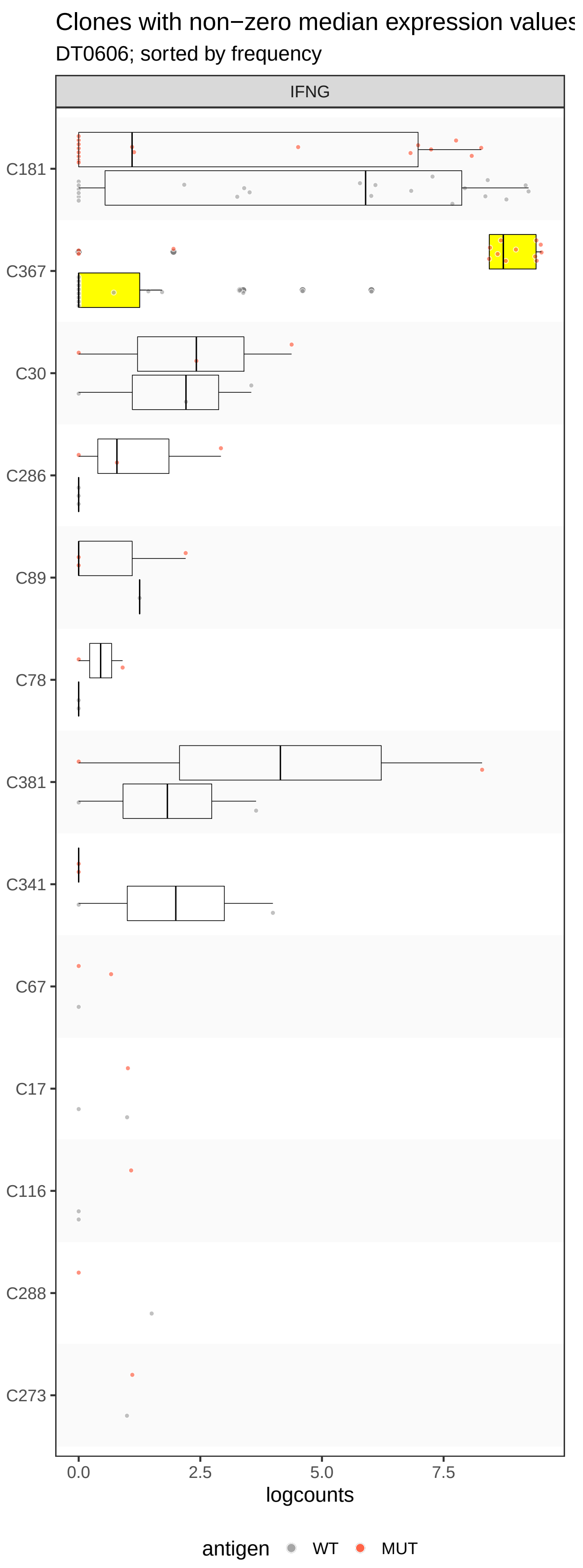
```r
t06  <- Klebanoff0606T1::prep_data4tgram(sce.06,
  which_clonotypes = unique(sce.06$id),
  DEresult_list = de.06, goi = "IFNG",
  additional_colData = c("antigen","freq_per_Sample", "freq_across_all",
    "fit_for_test", "Patient"))
setorder(t06, freq_across_all)
t06$id <- factor(t06$id, levels = unique(t06$id), ordered = TRUE)

pl <- list()
ymax <- ceiling(max(t06$logcounts))

for(x in unique(t06$gene_name)){
  tmp <- t06[gene_name == x]
  bp <- tmp[, median(logcounts), by = c("antigen","id")] %>% .[V1 > 0] %>%
    .$id %>% as.character %>% unique
  P <- ggplot(tmp[freq_across_all > 20], aes(x = id, y = logcounts))
  if(nrow(tmp[star != ""])>0){
    P <- P + geom_point(alpha = 0, aes(color = antigen)) +
      geom_tile(data=tmp[star != ""],
        aes(x = id, y = 1, height = Inf, width = 1),
        alpha = 1, fill = "khaki1")
  }
  P <- P +  ggbeeswarm::geom_quasirandom(
    groupOnX = TRUE, size = 1.5, shape = 21, alpha =.7,
      dodge.width = .8, aes(fill = antigen), color = "white") +
    facet_grid(.~gene_name) + coord_flip(ylim = c(0,ymax)) +
    geom_boxplot(data=tmp[freq_across_all > 3 & as.character(id) %in% bp],
      outlier.alpha = 0,lwd=.25, fill = NA, aes(color = antigen)) +
    theme(
      legend.position = "bottom",
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      panel.grid.major.y = element_line(size = .2)) +
    xlab("") +
  scale_fill_manual(values =  c("grey65","tomato1")) +
  scale_color_manual(values = c("black","black")) +
    guides(fill = guide_legend(override.aes = list(size = 3, alpha = 1) ),
      color = FALSE )
  pl[[x]] <- P
}

pw06 <- pl[[1]]
pw06 + plot_annotation(title =  "Clones with >20 cells across both antigens",
  subtitle = "DT0606")
```

Clones with >20 cells across both antigens

DT0606



```r
# For DT0606 it probably makes more sense to show the ones with non-zero median
# expression values in at least one gene
medsT06 <- t06[, median(logcounts), by = c("antigen","id","gene_name")] %>%
  .[V1 > 0] %>% .$id  %>% as.character %>% unique

tmp <- t06[as.character(id) %in% medsT06]
setorder(tmp, freq_across_all)
tmp$id <- factor(tmp$id, levels = unique(tmp$id), ordered = TRUE)
odd_numbers <- levels(tmp$id)[seq(1, nlevels(tmp$id),2)]

ggplot(tmp, aes(x =id, y = logcounts))+
  geom_point(alpha = 0, aes(fill = antigen)) + # decoy points to set up the x axis
  ## grey background for alternating IDs
    geom_tile(
      data = tmp[as.character(id) %in% odd_numbers],
      aes(x = id, y = 1, height = Inf, width = 1),
      alpha = 1, fill = "grey98") +
    ## highlight sign. ones
    geom_boxplot(data=tmp[star != ""],
      outlier.alpha = 0.5,lwd=.25, fill = "yellow",
      aes(color = antigen)) +
  ## plot the actual values per sample
    ggbeeswarm::geom_quasirandom(
      groupOnX = TRUE, size = 1.5, shape = 21, alpha =.7,
      dodge.width = .8, aes(fill = antigen), color = "white") +
  facet_wrap(~gene_name) + coord_flip() +
  geom_boxplot(
    data=tmp[freq_across_all > 3],
    outlier.alpha = 0,lwd=.25, fill = NA, aes(color = antigen)) +
  ## remove grind lines etc.
  theme(
      legend.position = "bottom",
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      panel.grid.major.y = element_blank()) +
  xlab("") +
  scale_fill_manual(values =  c("grey65","tomato1")) +
  scale_color_manual(values = c("black","black")) +
  guides(
    fill = guide_legend(override.aes = list(size = 3, alpha = 1) ),
    color = FALSE ) +
  ggtitle("Clones with non-zero median expression values",
    subtitle = "DT0606; sorted by frequency")
```



Clones with non−zero median expression values
DT0606; sorted by frequency
IFNG
C181
C367
C30
C286
C89
C78
C381
C341
C67
C17
C116
C288
C273
0.0
2.5
5.0
7.5
logcounts
antigen
WT
MUT

ExtDataFig 18: MR4050

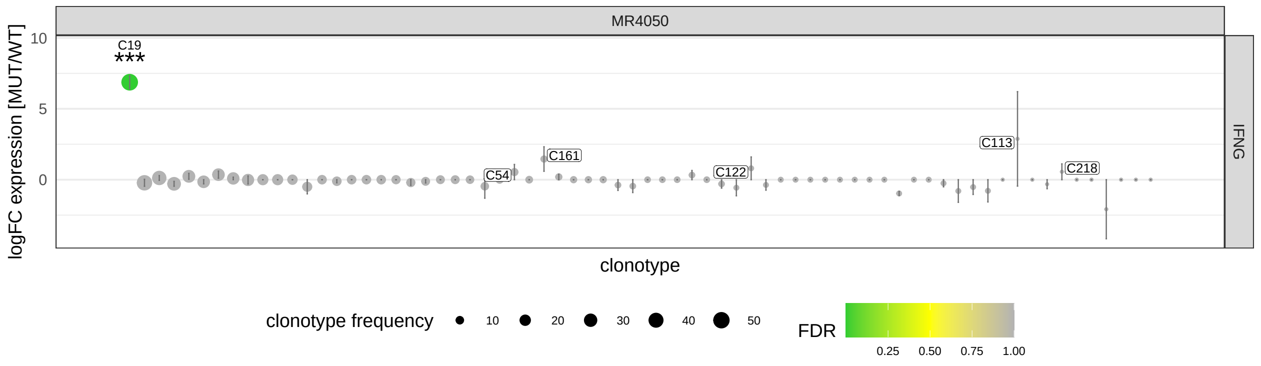
- for volcano plots and boxplots (C-E), see `DEgenes_MR4050.xlsx` and `MR4050_C19_Winner_logcounts_CD3-4-8.txt` – these plots were, in fact, done with GraphPad Prism

```
sce.mr <- KlebanoffMR4050::load_MR4050shared()
KlebanoffMR4050::load_DE_results()
de.mr <- delist.both

rm(delist.both); invisible(gc())

tmr  <- Klebanoff0606T1::prep_data4tgram(sce.mr, which_clonotypes = unique(sce.mr$id),
  DEresult_list = de.mr, goi = "IFNG",
  additional_colData = c("antigen","freq_per_Sample",
    "freq_across_all","fit_for_test", "Patient"))

# MR4050 ====================================
## calculate CI etc.
testMRres <- tmr[
  fit_for_test == TRUE ,
  extract.tres(t.test(logcounts~antigen)),
  by = c("id", "gene_name")]

## summarize
tMR.summ <- tmr[, -c("cell", "cdr3s_aa","logcounts", "Sample","antigen",
  "freq_per_Sample"), with=FALSE] %>%  unique %>% .[testMRres, on = "id"] %>%
  unique
tMR.summ[ is.na(ci.low), ci.low:=0]
tMR.summ[ is.na(ci.up), ci.up:=0]
```

- Error bar: SE

```
tMR.summ0 <- copy(tMR.summ)

## if delta IFNG < 0, set to 0.
tMR.summ0[, logFC := ifelse(logFC.t > 0, logFC.t, 0)]
tMR.summ0[, ci.low0 := ifelse(logFC.t >0, ci.low, 0)]
tMR.summ0[, ci.up0 := ifelse(logFC.t >0, ci.up, 0)]

## sort by frequency
setorder(tMR.summ0, -freq_across_all)
tMR.summ0[, comb.ID := paste(id, Patient, sep = ".")]
tMR.summ0$id.sort <- factor(tMR.summ0$comb.ID,
  levels = tMR.summ0$comb.ID, ordered = TRUE)

tMR.summ0[is.na(FDR), FDR := 1]
tMR.summ0[, neg.log10.FDR := -1*log10(FDR)]
```

```
ystar <- max(tMR.summ0$logFC.t) + 1.5
ggplot(tMR.summ0, aes(x = id.sort, y = logFC.t)) +
  geom_point(aes(size = freq_across_all, color = FDR)) +
  geom_errorbar(aes(ymin=logFC.t-sde, ymax=logFC.t+sde), width=.1, color="grey45") +
  facet_grid(gene_name ~ Patient, scales="free_x", space = "free_x") +
  ## add asterisks -----------------------
  geom_text(inherit.aes = FALSE,
    aes(x = id.sort, y = ystar, label=star),
    colour="black", size=8)+
  geom_text(data = tMR.summ0[star != ""],
    inherit.aes = FALSE,
    aes(x = id.sort, label = id, y = 9.5)) +
  ## grid appearance etc. ----------------
  theme(panel.grid.minor.x=element_blank(),
    panel.grid.major.x=element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_blank(),
    legend.position = "bottom",
    legend.text = element_text( size = 10),
    legend.key.size = unit(2,"line")) +
  scale_color_gradientn( colours = rev(c("gray70","yellow","limegreen"))) +
  scale_size(name = "clonotype frequency") +
  xlab("clonotype") + ylab("logFC expression [MUT/WT]") +
  scale_x_discrete( expand = expansion(add = 5)) +
  ggrepel::geom_label_repel(
    data = tMR.summ0[logFC.t > 0.5 & id!="C19"],
    box.padding = 0.1, label.padding = 0.1, label.size = 0.1,
    aes(label = id))
```


MR4050
IFNG
C19
***
C161
C54
C122
C113
C218
logFC expression [MUT/WT]
clonotype
clonotype frequency
FDR