xv20230329_RPE1_pool_pathway assay

Xabier Vergara

2023-07-26

#Date: 26th July 2023 #Author: Xabier Vergara #Aim: The aim of this file is to wrap up the filtering strategy and measure if pathway balance in control samples has chromatin biases.

#Load libraries

Import data

```
# List files in mcclintock
pathway_assay_files <- list.files("/DATA/projects/DSBrepair/data/xv20230714_E2286_pool_pathway_assay/in

#Read and parse files
clone_pools_pathway_assay <- map_dfr(pathway_assay_files, function(x){
  read_delim(x, show_col_types = F) %>%
  mutate(filename = gsub("/DATA/projects/DSBrepair/data/xv20230714_E2286_pool_pathway_assay/indelPCR_co
         outcome = paste(call, indel, sep = "_")) %>%
  separate(filename, into = c("exp_n","sample_ID","cell_line","condition","bio_rep","t_rep"))
})
```

#Number of mapped IPRs

```
selected_pools <- c("RPE1_Low_1000","RPE1Deff_Low_1000","RPE1Proff_Low_250","U2OS_High_100")

#Import barcode sequences
#RPE cells
RPE_IPR <- read_delim("/home/x.vergara/XV_P3_ChromDSBScreen/xv_CRISPR_screen_figures/export/xv20230307_
  select(name) %>%
  separate(name, into = c("barcode","iPCR_cell","iPCR_transfection","iPCR_complexity"))
```

```
## Rows: 4314 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (2): name, chr
## dbl (1): start
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
U2OS_IPR <- read_delim("/home/x.vergara/XV_P3_ChromDSBScreen/xv_CRISPR_screen_figures/export/xv20230307
  select(name) %>%
  separate(name, into = c("barcode","iPCR_cell","iPCR_transfection","iPCR_complexity"))
```

```
## Rows: 2015 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
```

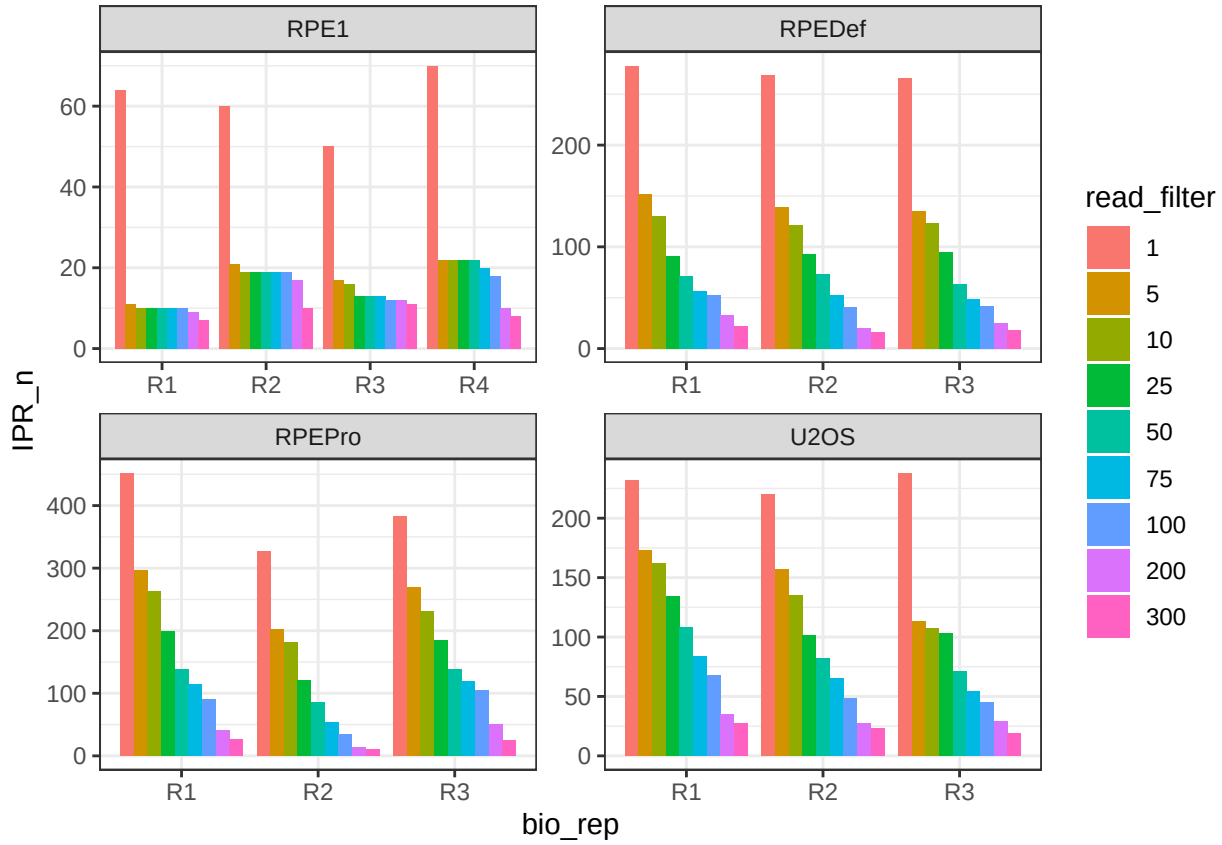
```
## chr (2): name, chr
## dbl (1): start
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#Bind both
all_IPRs <- bind_rows(RPE_IPR, U2OS_IPR) %>% mutate(pool = paste(iPCR_cell, iPCR_transfection,iPCR_compl
```

#Test different filtering strategies: How many reads you need for reliable pathway balance measurement. I decided that I will do this by filter x amount of reads of -7_del and 1_ins. I tried to do this by total amount of reads, but then the readout becomes very noisy in the sense that there are some IPRs where I have 1000 reads of ins_1 and 0 del_-7. This also happens the other way around. NOTE: I will also add a pseudocount here, this is not necessary for this first analysis, but will be needed for ATM and DNAPKi comparison.

```
#Filter by replicate
indel_data <- clone_pools_pathway_assay %>%
  dplyr::group_by(barcode,outcome, exp_n, cell_line, condition, bio_rep) %>%
  dplyr::summarise(counts = sum(count, na.rm = F) +1) %>% left_join(all_IPRs) %>% na.omit()

#Filter out rep (This replicate doesn't correlate with the rest)
filter_out_rep <- tibble(cell_line = "RPEDef",bio_rep = "R4")
indel_data_clean <- indel_data %>% anti_join(filter_out_rep)


#Only look at LBR2 vs. tracr
control_balance_bio <- indel_data_clean %>% filter(condition %in% c("LBR2","tracr") & outcome %in% c("i

#Look at read numbers
control_balance_bio_dcast <- control_balance_bio %>% reshape2::dcast(barcode + cell_line + condition + l

#Control_measurements (bio_rep)
pathway_metric_control_bio <- control_balance_bio_dcast %>%
  mutate(balance = `del_-7`/(ins_1 + `del_-7`),
         log2_bal = log2(`del_-7`/ins_1),
         TIF = 1 - (wt_0/(`del_-7`+ins_1+wt_0))) %>%
  na.omit()

#Filter by read number and plot noise level
read_count <- c(1,5,10,25,50,75,100,200,300)

filter_read_count_test <- map_dfr(read_count , function(x) {
  pathway_metric_control_bio %>%
    filter(`del_-7` >= x & ins_1 >= x) %>%
    mutate(read_filter = x)
})

#Number of IPRs in each
IPR_summary <- filter_read_count_test %>%
  dplyr::group_by(cell_line, condition, read_filter, bio_rep) %>%
  dplyr::summarize(IPR_n = n())

IPR_summary_dcast <- IPR_summary %>%
  reshape2::dcast(cell_line + condition + bio_rep ~ read_filter, value.var  = "IPR_n")
```

```
#Plot loss of IPRs by each read_count filter

ggplot(IPR_summary %>%
         filter(condition ==  "LBR2")) +
  geom_col(aes(x = bio_rep,
               y = IPR_n,
               fill = fct_relevel(as.character(read_filter),as.character(read_count))),
           position = "dodge") +
  labs(fill = "read_filter") +
  facet_wrap(~ cell_line, scales = "free") +
  theme_bw()
```



#Conclusion: In RPE1 cells there seem to be ~20 reporters that are always there. The rest of cell line we gradually lose IPRs as we increase the filtering strength

#Do I find the same IPRs in all replicates? Or these are different from replicate to replicate.

```
#Calculate co-ocurrence per replicate
replicates_IPR <- filter_read_count_test %>%
  filter(condition == "LBR2") %>%
  dplyr::group_by(barcode,cell_line, read_filter) %>%
  dplyr::summarise(rep_n = n()) %>%
  ungroup() %>%
  dplyr::group_by(cell_line, read_filter,rep_n) %>%
  dplyr::summarise(capture_IPR = n())
```

```
## `summarise()` has grouped output by 'barcode', 'cell_line'. You can override
## using the `.groups` argument.
```

```
## `summarise()` has grouped output by 'cell_line', 'read_filter'. You can
## override using the `.groups` argument.
```

```
total_IPR_rep <- filter_read_count_test %>%
  filter(condition == "LBR2") %>%
  select(cell_line, barcode, read_filter) %>%
  distinct() %>%
  dplyr::group_by(cell_line,read_filter) %>%
  dplyr::summarise(total_IPR = n())
```

```
## `summarise()` has grouped output by 'cell_line'. You can override using the
## `.groups` argument.
```

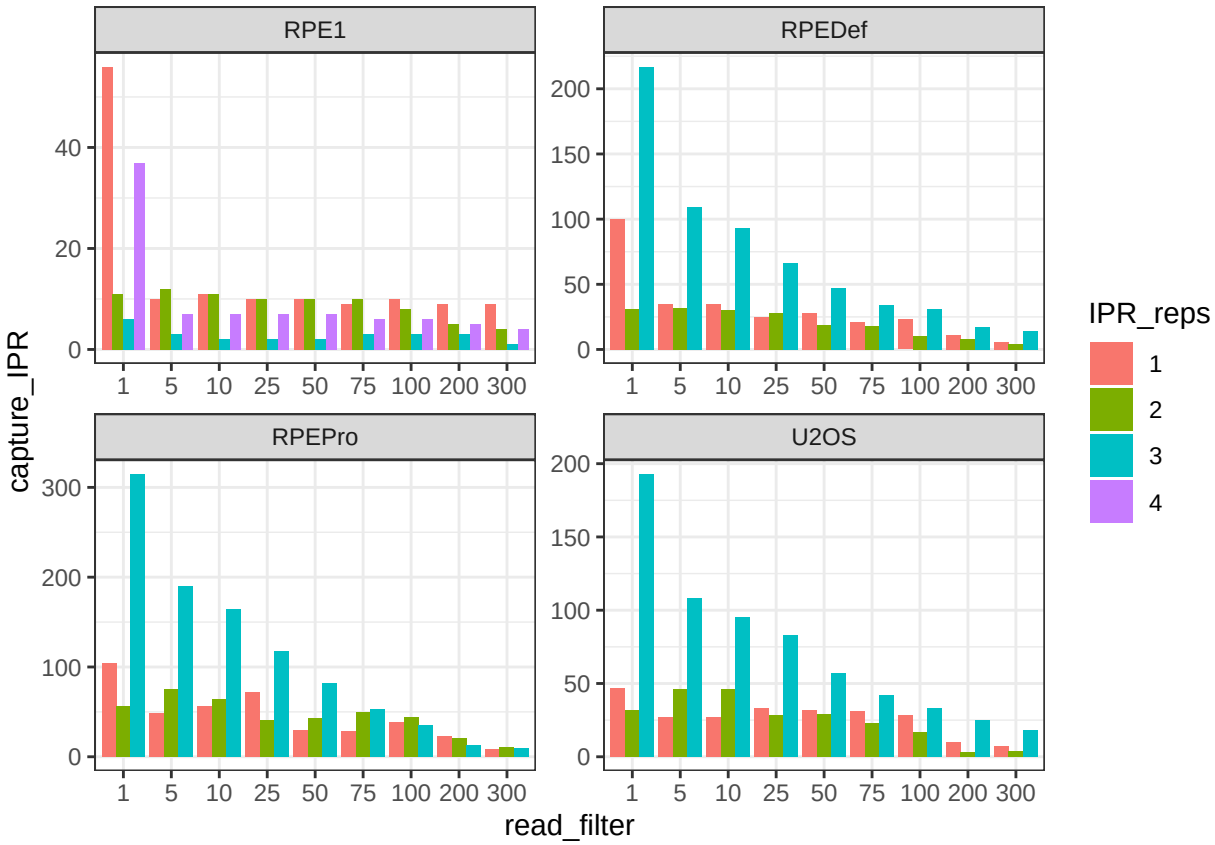
```
#Calculate frequency
freq_IPR_capture <- replicates_IPR %>%
  left_join(total_IPR_rep) %>%
  mutate(freq_capture = capture_IPR/total_IPR)
```

```
## Joining, by = c("cell_line", "read_filter")
```

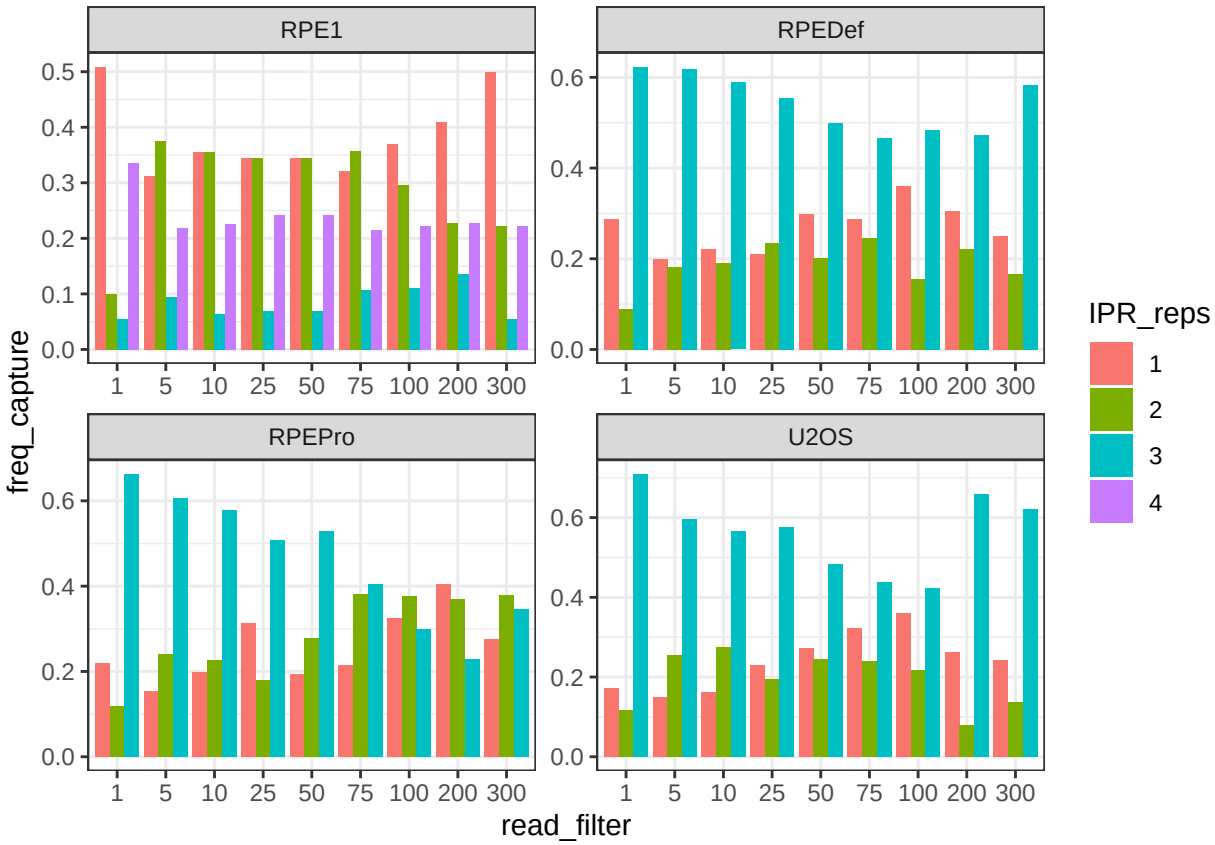
```
#Absolute_numbers
IPR_reps <- replicates_IPR %>% reshape2::dcast(cell_line + rep_n ~ read_filter)
```

```
## Using capture_IPR as value column: use value.var to override.
```

```
#IPRs in multiple reps
ggplot(replicates_IPR) +
  geom_col(aes(x = fct_relevel(as.character(read_filter),as.character(read_count)),
               y = capture_IPR,
               fill = as.character(rep_n)),
           position = "dodge") +
  labs(fill = "IPR_reps") +
  facet_wrap(~ cell_line, scales = "free") +
  theme_bw() + labs(x = "read_filter")
```




```
#Frequency
ggplot(freq_IPR_capture) +
  geom_col(aes(x = fct_relevel(as.character(read_filter),as.character(read_count)),
               y = freq_capture,
               fill = as.character(rep_n)),
           position = "dodge") +
  labs(fill = "IPR_reps") +
  facet_wrap(~ cell_line, scales = "free") +
  theme_bw() + labs(x = "read_filter")
```



#Conclusion: Data in RPE1 is very sparce

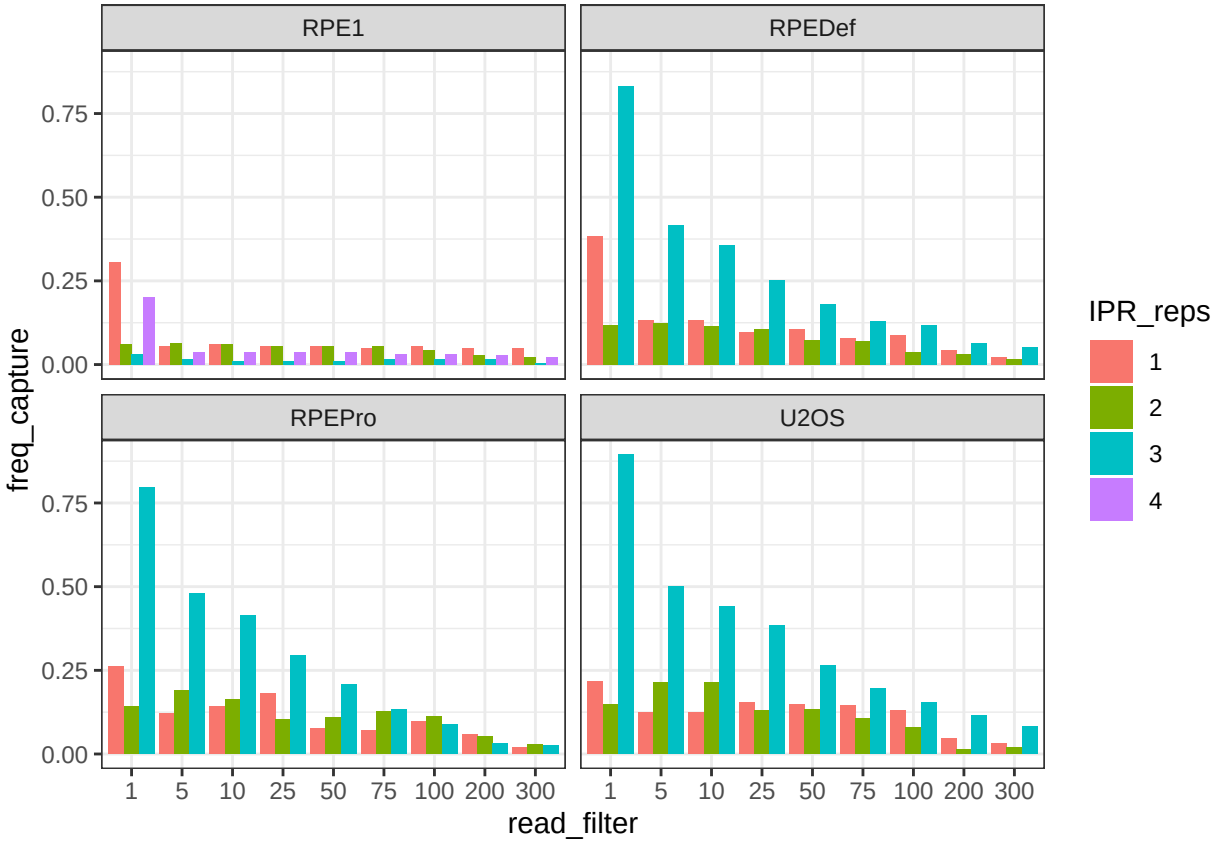
#Calculate frequency in absolute number of IPRs

```
#Total mapped IPR per pool
summary_mapped <- all_IPRs %>%
  dplyr::group_by(iPCR_cell) %>%
  dplyr::summarise(total_IPR = n()) %>%
  mutate(cell_line = case_when(iPCR_cell == "RPE1Deff" ~ "RPEDef",
                               iPCR_cell == "RPE1Proff" ~ "RPEPro",
                               T ~ iPCR_cell)) %>%
  select(cell_line, total_IPR)

#Calculate frequency out of all IPRs mapped
freq_IPR_capture_total <- replicates_IPR %>%
  left_join(summary_mapped) %>%
  mutate(freq_capture = capture_IPR/total_IPR)
```

```
## Joining, by = "cell_line"
```

```
#Frequency
ggplot(freq_IPR_capture_total) +
  geom_col(aes(x = fct_relevel(as.character(read_filter),as.character(read_count)),
               y = freq_capture,
               fill = as.character(rep_n)),
           position = "dodge") +
  labs(fill = "IPR_reps") +
  facet_wrap(~ cell_line) +
```

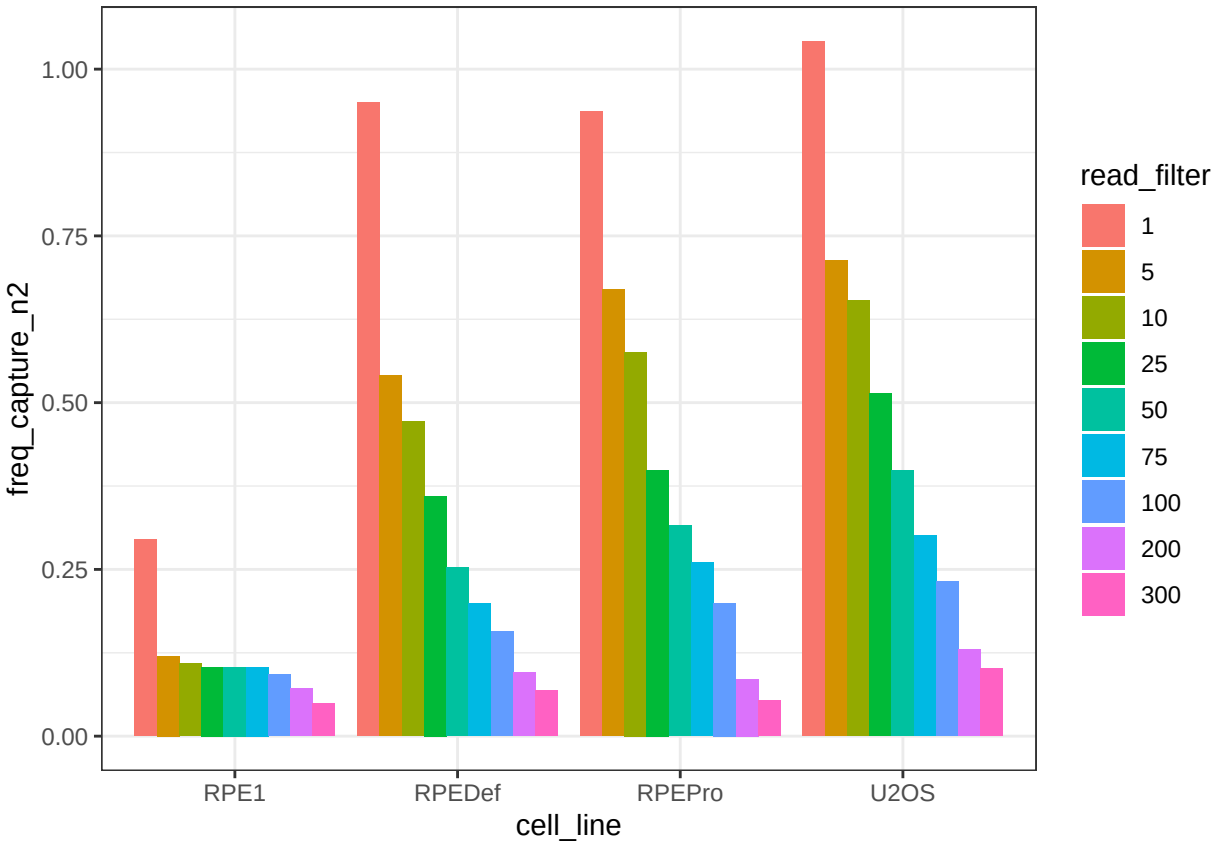
```
  theme_bw() + labs(x = "read_filter")
```




```
#How many IPRs have at least n=2
n2_IPRs <- freq_IPR_capture_total %>%
  filter(rep_n > 1) %>%
  dplyr::group_by(cell_line, read_filter) %>%
  dplyr::summarise(IPR_n = sum(capture_IPR)) %>%
    left_join(summary_mapped) %>%
  mutate(freq_capture_n2 = IPR_n/total_IPR)
```

```
## `summarise()` has grouped output by 'cell_line'. You can override using the
## `.groups` argument.
## Joining, by = "cell_line"
```

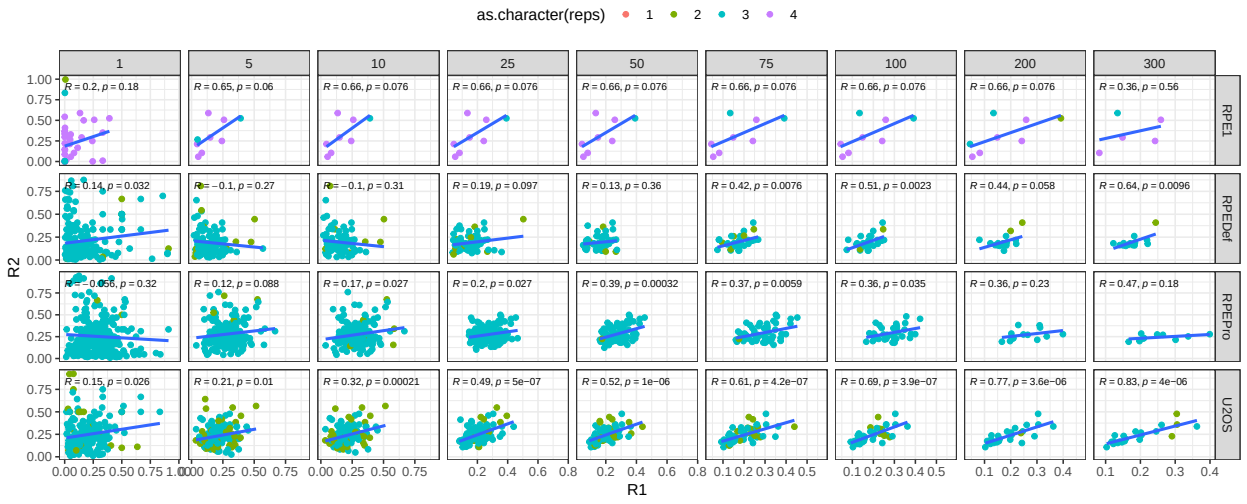
```
#Frequency
ggplot(n2_IPRs) +
  geom_col(aes(x = cell_line,
               y = freq_capture_n2,
               fill = fct_relevel(as.character(read_filter),as.character(read_count))),
           position = "dodge") +
  labs(fill = "read_filter") +
  theme_bw()
```



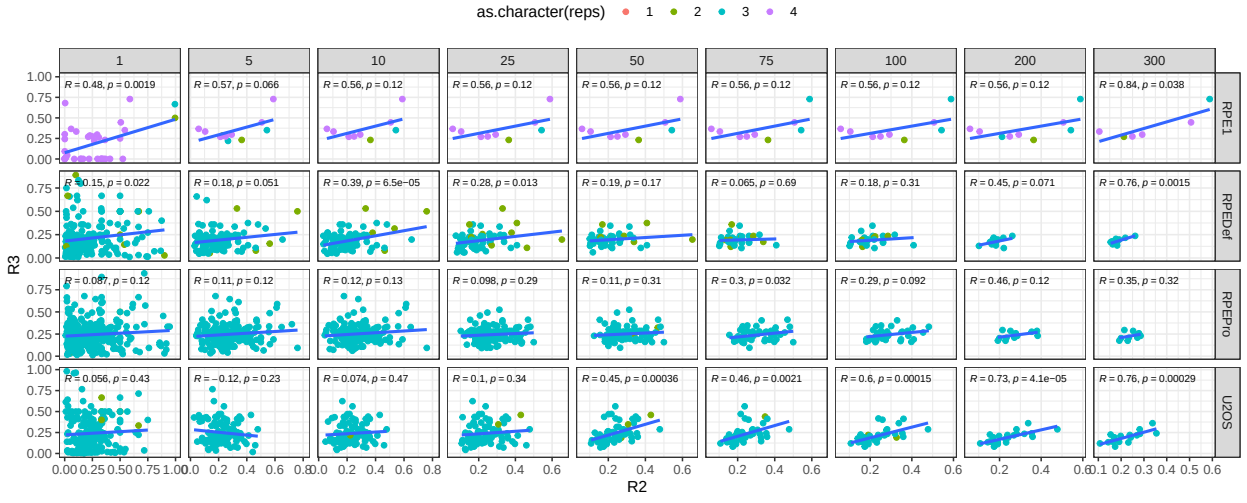
#Conclusion: We capture, with at least n=2, a good proportion of all pools in DEF, PRO and U2OS. But, this is much lower in RPE1 cells.

#Is pathway balance reproducible among replicates?

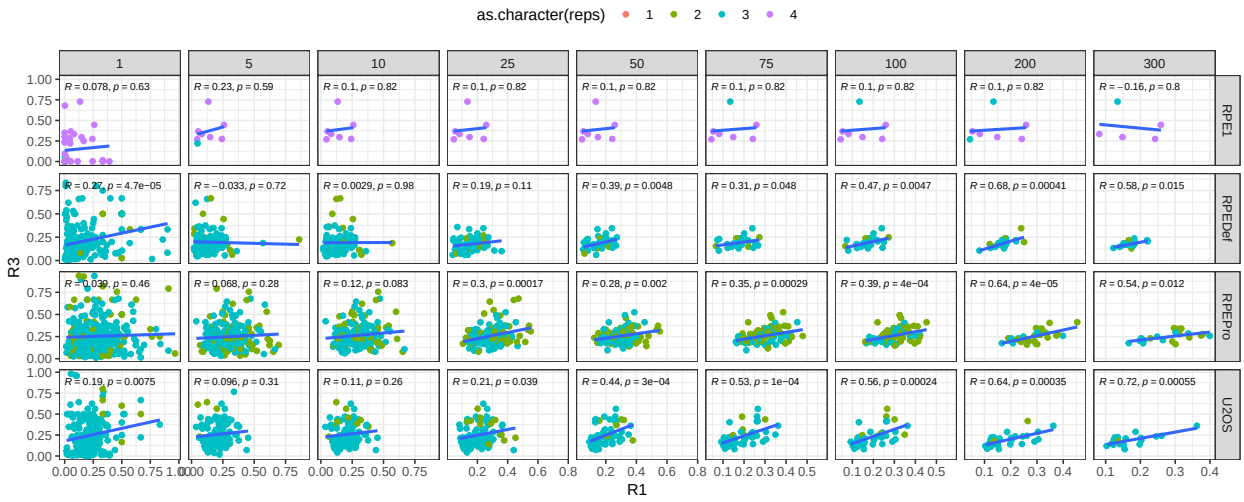
```
read_counts_filter_reprod <- filter_read_count_test %>%
  filter(condition == "LBR2") %>%
  reshape2::dcast(barcode + cell_line + condition + read_filter ~ bio_rep, value.var = "balance")

#Calculate mean & sd
read_counts_filter_reprod_summary <- read_counts_filter_reprod %>%
  rowwise() %>%
  mutate(mean_bal = mean(c(R1,R2,R3,R4),na.rm = T),
         sd_bal = sd(c(R1,R2,R3,R4), na.rm = T),
         reps = 4 - sum(is.na(c(R1,R2,R3,R4))))

#R1 vs. R2
ggplot(read_counts_filter_reprod_summary, aes(R1,R2)) +
  geom_point(aes(color = as.character(reps))) +
  geom_smooth(method = "lm", se = F) +
  ggpubr::stat_cor(method = "pearson",size = 2.5) +
  facet_grid(cell_line ~ read_filter, scales = "free") +
  theme_bw() +
  theme(legend.position = "top") + labs(fill = "replicate number")
```




```
#R1 vs. R2
ggplot(read_counts_filter_reprod_summary, aes(R2,R3)) +
  geom_point(aes(color = as.character(reps))) +
  geom_smooth(method = "lm", se = F) +
  ggpubr::stat_cor(method = "pearson",size = 2.5) +
  facet_grid(cell_line ~ read_filter, scales = "free") +
  theme_bw() +
  theme(legend.position = "top") + labs(fill = "replicate number")
```




```
#R1 vs. R2
ggplot(read_counts_filter_reprod_summary, aes(R1,R3)) +
  geom_point(aes(color = as.character(reps))) +
  geom_smooth(method = "lm", se = F) +
  ggpubr::stat_cor(method = "pearson",size = 2.5) +
  facet_grid(cell_line ~ read_filter, scales = "free") +
  theme_bw() +
  theme(legend.position = "top") + labs(fill = "replicate number")
```



#Conclusion: The correlation improves as we increase the amount of reads, but the measurements are quite noisy. The RPEDef have the worst correlation score.

#Plot summary correlation coefficients

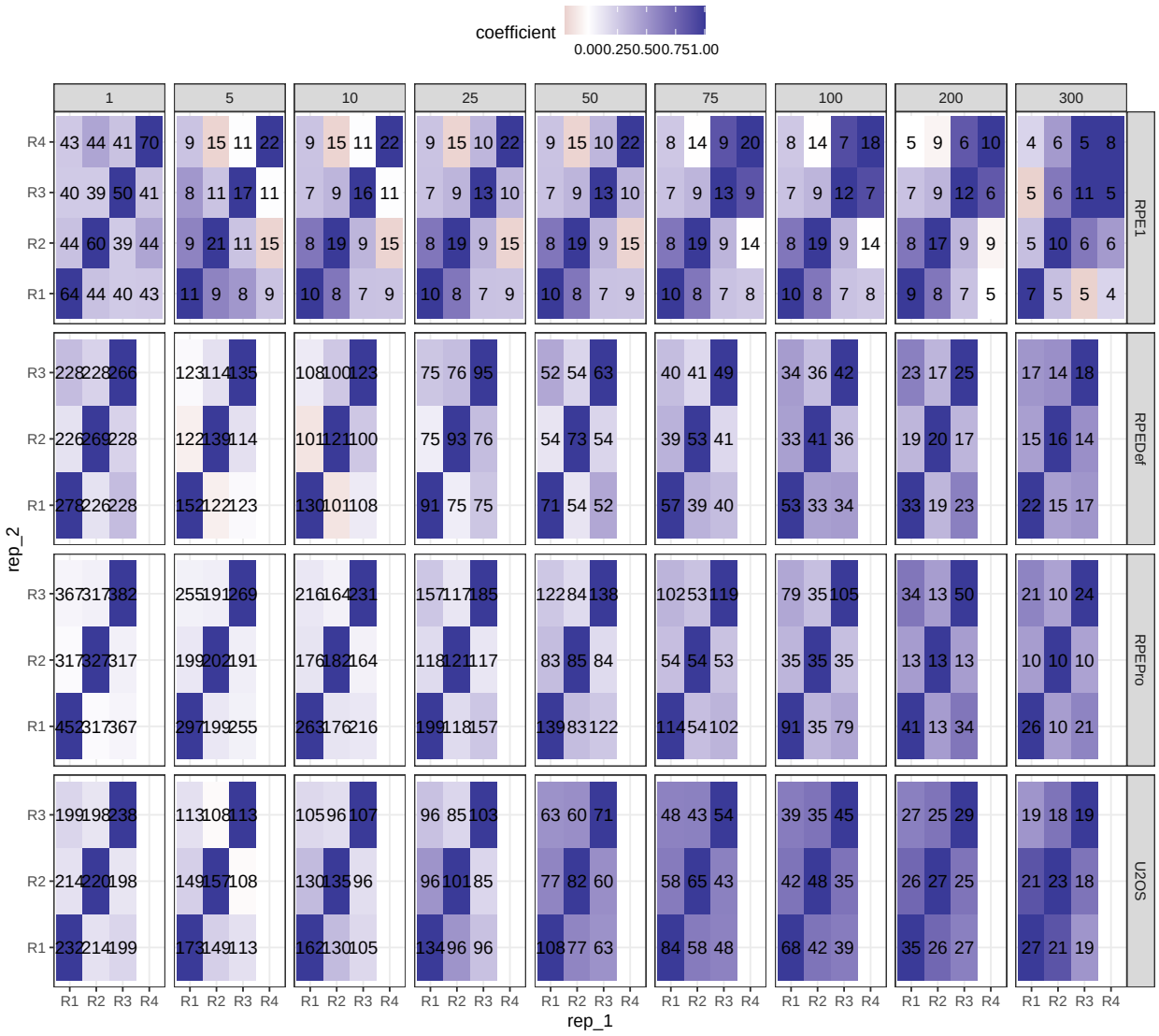
```
#Test with U2OS & 50
analysis_pair <- read_counts_filter_reprod_summary %>% select(cell_line, read_filter) %>% distinct()

replicate_comparisons <- tidyr::crossing(c(1:4),c(1:4))

tmp_f <- map2_dfr(analysis_pair$cell_line,analysis_pair$read_filter, function(x,y){
  #Filter_data
  data_sel <- read_counts_filter_reprod_summary %>%
    filter(cell_line == x & read_filter == y) %>%
    select(R1, R2, R3, R4)

  #Correlation analyis
cor_data <- cor(data_sel, use = "pairwise.complete.obs", method = "spearman") %>%
  as_tibble(rownames = NA) %>%
  rownames_to_column(var = "rep_1") %>%
  reshape2::melt(value.name = "coefficient") %>%
  select(rep_1, "rep_2" = variable, coefficient) %>%
  na.omit() %>%
  mutate(cell_line = x,
         read_filter = y)

#Number of IPRs per each data table

IPR_n_cor <- map2_dfr(as.vector(as.matrix(replicate_comparisons[1])), as.vector(as.matrix(replicate_comp
  IPR_count <- data_sel %>% select(j,i) %>% na.omit() %>% nrow()
  tibble(rep_1 = paste0("R",j), rep_2 = paste0("R",i), IPR_n = IPR_count)
})

left_join(cor_data, IPR_n_cor)

})
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
```

```
##   # Was:
##   data %>% select(j)
##
##   # Now:
##   data %>% select(all_of(j))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(i)
##
##   # Now:
##   data %>% select(all_of(i))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
#Plot with data_points in comparison
ggplot(tmp_f) +
  geom_tile(aes(rep_1, rep_2, fill = coefficient)) +
  geom_text(aes(rep_1, rep_2, label = IPR_n)) +
  scale_fill_gradient2() +
  facet_grid(cell_line ~ read_filter, scales = "free") +
  theme_bw() +
  theme(legend.position = "top")
```



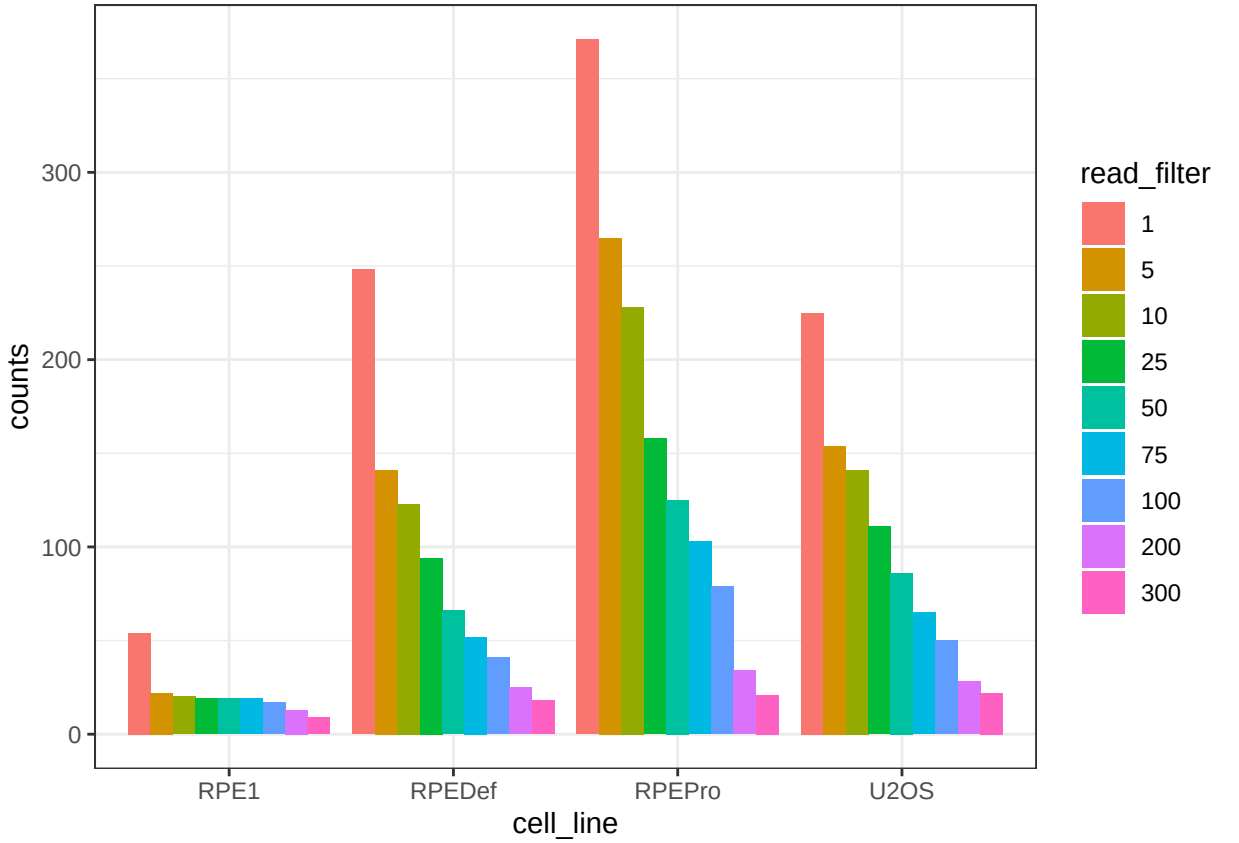
#How does data look in this filter

```
#Balance data
filtered_balance <- read_counts_filter_reprod_summary %>%
  filter(reps > 1) %>%
  select(barcode, cell_line, mean_bal, sd_bal, reps, read_filter)

#Number of IPRs per cell line and read_filter
IPR_number <- filtered_balance %>% dplyr::group_by(cell_line,read_filter) %>% dplyr::summarise(counts =
```

```
## `summarise()` has grouped output by 'cell_line'. You can override using the
## `.groups` argument.
```

```
ggplot(IPR_number) +
  geom_col(aes(cell_line,
               counts,
               fill = fct_relevel(as.character(read_filter), as.character(read_count))),
           position = "dodge") +
  theme_bw() +
```

```
  labs(fill = "read_filter")
```




```
#I will filter on 75 reads
filtered_balance_75 <- filtered_balance %>% filter(read_filter == 75)
```

#Add chromatin info for RPE1 cells #Load chromatin info of the IPRs in RPE cells

```
setwd("/DATA/projects/DSBrepair/data/DSB_TRIP_cell_lines_mapping/xv20230519_all_RPE_pools_mapping/")
# Load chip data
chip_files <- list.files(path = "chip/site_means", pattern = "2000", full.names = T)
dam_files <- list.files(path = "dam", pattern = "2000", full.names = T)
repli_files <- "coverage/RPE1_DSB_TRIP_pools-2000_summary.tsv"

#Make a dataframe with all the files
chromatin_features_pools_chip <- map(chip_files, function(x) {
  read.delim(x, header = T) %>%
    dplyr::select(ID, z_score) %>%
    na.omit() %>%
    mutate(file = paste0("chip_",str_extract(x, "H.*(?=_)"))) %>%
    reshape2::dcast(ID ~ file, value.var = "z_score")
}) %>% purrr::reduce(left_join, by = "ID")

#Make a dataframe with dam files
chromatin_features_pools_dam <- map(dam_files, function(x) {
  read.delim(x, header = T) %>%
    dplyr::select(name, z_score) %>%
    na.omit() %>%
```

```
    mutate(file = paste0("dam_",str_extract(x, "(?<=2000_).*(?=.txt)"))) %>%
    reshape2::dcast(name ~ file, value.var = "z_score")
}) %>% purrr::reduce(left_join, by = "name")

#Make a column for repliseq
repliseq_dataframe <- map_dfr(repli_files, function(x) {
  read.delim(x, header = T) %>%
    na.omit() %>%
    mutate(ratio_r1 = repliseq_r1_late - repliseq_r1_early, ratio_r2 = repliseq_r2_late - repliseq_r2_ea
    mutate(repliseq = (ratio_r1 + ratio_r2)/2) %>%
    dplyr::select("ID" = barcode, repliseq)
})

#Bind both
chromatin_features_pools <-  left_join(chromatin_features_pools_chip, chromatin_features_pools_dam, by =
  left_join(repliseq_dataframe) %>%
  separate(ID, into = c("barcode","iPCR_cell","iPCR_transfection","iPCR_complexity")) %>%
  mutate(pool = paste(iPCR_cell, iPCR_transfection,iPCR_complexity, sep = "_")) %>%
  filter(pool %in% selected_pools) %>%
  mutate(cell_line = case_when(iPCR_cell == "RPE1Deff" ~ "RPEDef",
                               iPCR_cell == "RPE1Proff" ~ "RPEPro",
                               T ~ iPCR_cell)) %>%
  select(-pool, -iPCR_cell, -iPCR_transfection, -iPCR_complexity)
```

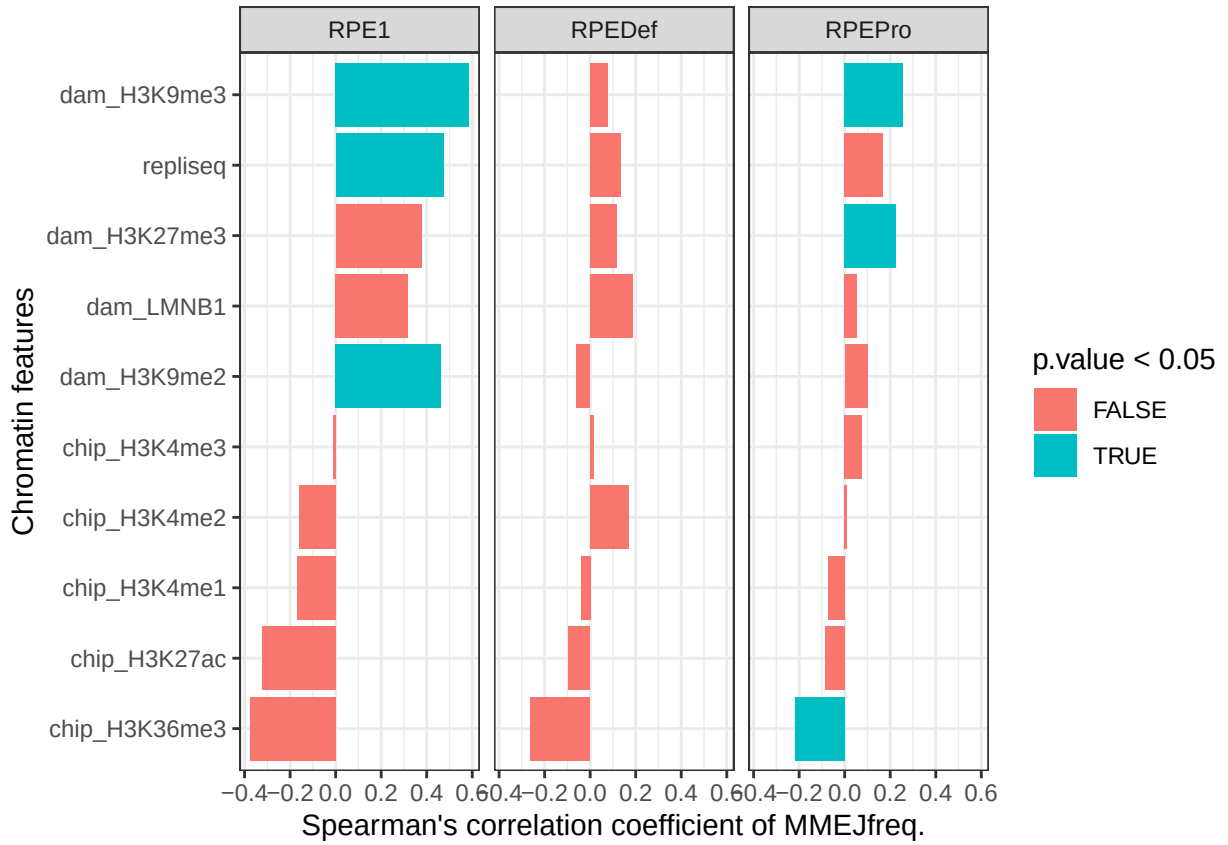
```
## Joining, by = "ID"
```

#Join with balances and calculate the correlation (do this for all marks)

```
values_chromatin <- left_join(filtered_balance_75, chromatin_features_pools) %>% filter(cell_line != "U
```

```
## Joining, by = c("barcode", "cell_line")
```

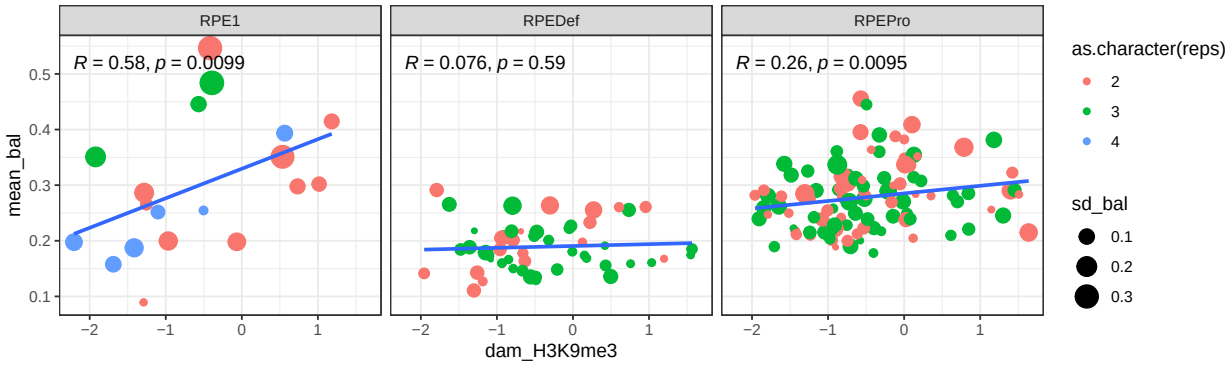
```
#Calculate spearman correlation for all chromatin features
chromatin_analysis_pair <- analysis_pair %>% filter(cell_line != "U2OS")

spearman_coefficients <- map_dfr(chromatin_analysis_pair$cell_line, function(x){
  filter_data <- values_chromatin %>% filter(cell_line == x)
  rho_table <- map_dfr(colnames(values_chromatin[7:16]), function(j){
      balance <- filter_data$mean_bal
      feature <- as.numeric(as.matrix(filter_data[j]))
      cor.test(x = balance, y = feature, method = "spearman", use = "pairwise.complete.obs") %>% broom:

})
  rho_table %>% mutate(cell_line = x) %>% distinct()
})

#Plot correlation coefficients
ggplot(spearman_coefficients %>% distinct()) +
  geom_col(aes(reorder(chromatin,estimate),estimate, fill = p.value < 0.05)) +
  coord_flip() +
  theme_bw() +
  xlab("Chromatin features") +
  ylab("Spearman's correlation coefficient of MMEJfreq.") +
  facet_grid(~ fct_relevel(cell_line, c("RPE1","RPEDef","RPEPro")))
```



#Conclusion: Filtering at 75 reads gives us a very similar mutation biases in RPE1 wild types. RPE p53null & RPE p53/BRCA1 null cells the biases are not so clear.

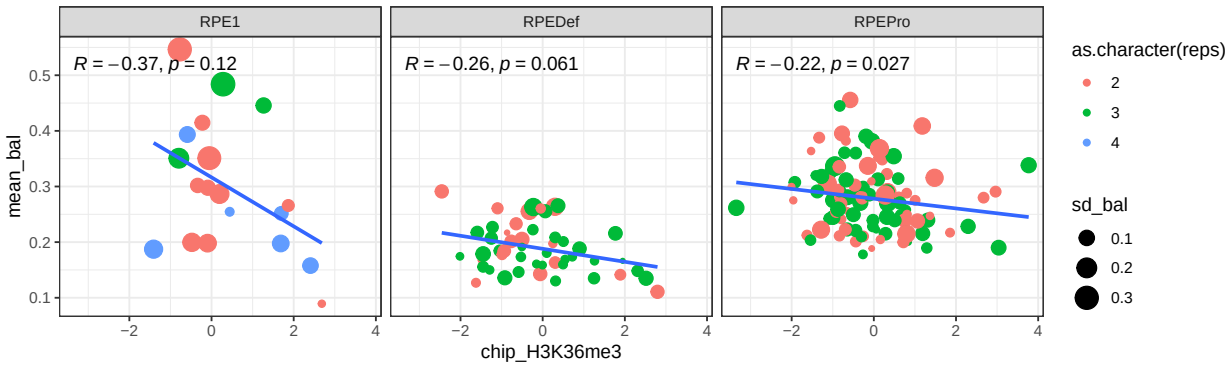
#Plot individual correlations: To check how they look

```
#Plot correlations (These are significant correlations)
ggplot(values_chromatin, aes(dam_H3K9me3, mean_bal)) +
  geom_point(aes(size =sd_bal, color = as.character(reps))) +
  facet_grid(~ cell_line) +
  geom_smooth(method = "lm",se = F)+
  ggpubr::stat_cor(method = "spearman") +
  theme_bw()
```




```
#Plot correlations (These are significant correlations)
ggplot(values_chromatin, aes(chip_H3K36me3, mean_bal)) +
```

```
  geom_point(aes(size =sd_bal, color = as.character(reps))) +
  facet_grid(~ cell_line) +
  geom_smooth(method = "lm",se = F)+
  ggpubr::stat_cor(method = "spearman") +
  theme_bw()
```

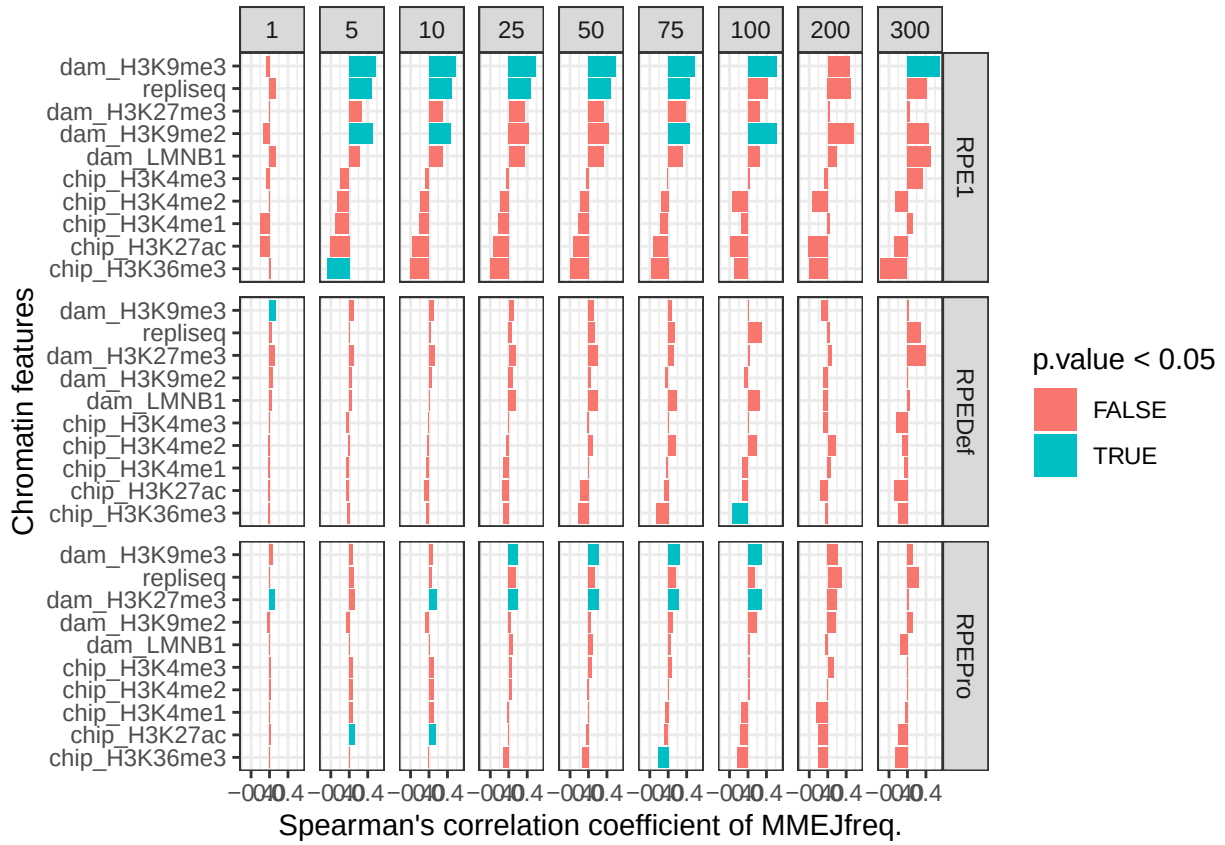


#As a control: are these biases consistent at different filtering steps

```
values_chromatin_all_filter <- left_join(filtered_balance, chromatin_features_pools) %>% filter(cell_li
```

```
## Joining, by = c("barcode", "cell_line")
```

```
#Calculate spearman correlation for all chromatin features
chromatin_analysis_pair <- analysis_pair %>% filter(cell_line != "U2OS")

spearman_coefficients_all <- map2_dfr(chromatin_analysis_pair$cell_line, chromatin_analysis_pair$read_f
  filter_data <- values_chromatin_all_filter %>% filter(cell_line == x & read_filter == y)
  rho_table <- map_dfr(colnames(values_chromatin_all_filter[7:16]), function(j){
      balance <- filter_data$mean_bal
      feature <- as.numeric(as.matrix(filter_data[j]))
      cor.test(x = balance, y = feature, method = "spearman", use = "pairwise.complete.obs") %>% broom:

})
  rho_table %>% mutate(cell_line = x, read_filter = y) %>% distinct()
})

#Plot correlation coefficients
ggplot(spearman_coefficients_all %>% distinct()) +
  geom_col(aes(reorder(chromatin,estimate),estimate, fill = p.value < 0.05)) +
  coord_flip() +
  theme_bw() +
  xlab("Chromatin features") +
  ylab("Spearman's correlation coefficient of MMEJfreq.") +
  facet_grid(fct_relevel(cell_line, c("RPE1","RPEDef","RPEPro")) ~ read_filter)
```



#Spearman correlation over different filters

```
ggplot(spearman_coefficients_all %>% distinct()) +
  geom_col(aes(fct_relevel(as.character(read_filter), as.character(read_count)), estimate, fill = cell_
  theme_bw() +
  xlab("Read filter") +
  ylab("Spearman's correlation coefficient of MMEJfreq.") +
  facet_grid(fct_relevel(cell_line, c("RPE1","RPEDef","RPEPro")) ~ chromatin) + theme(legend.position =
```