Home work 3

The data

In the directory you got with this file, there is a set of salmon-mapped mouse RNA-seq data. These are the same data as we worked with in the in-class exercises, with two important differences:

1: We have the same skin cell WT RNA-seq data as before (called WT1-3), but now also skin cells from six treated samples - 3 WT mice that have been have been treated with TPA, a cancer-inducing agent (called WTTPA1-3), and 3 mice which have the oncogene Kras knocked out (KO1-3).

2: These RNA-seq reads have been mapped using salmon to an 'internal' transcript index, so the transcript names cannot be found in e.g. the UCSC browser. Salmon was run with the –seqBias option.

Question 1: Effective lengths?

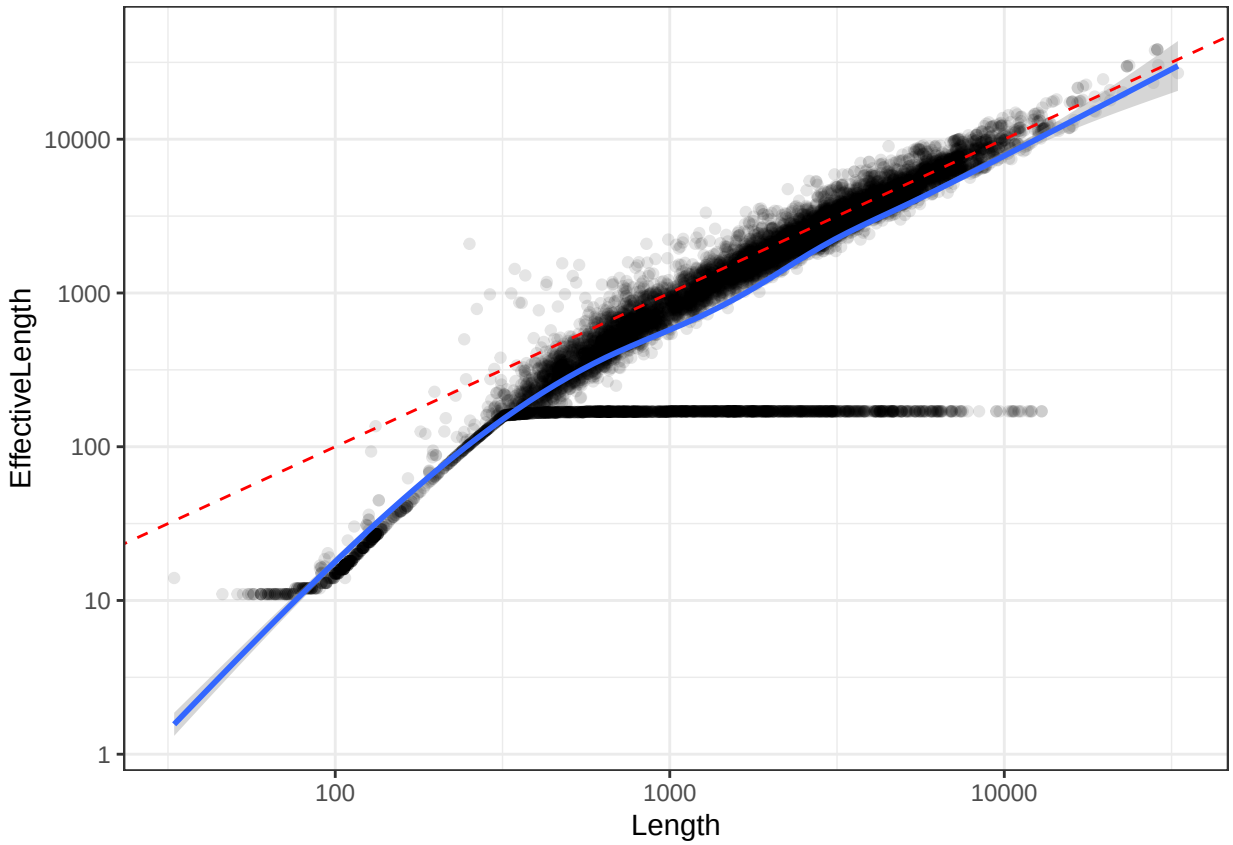
Using tidyverse only, read the `quant.sf` file from the WT1 Salon result folder into R. Plot the transcript annotated length versus effective length for transcripts with any detected expression in an xy plot with log10 scaled x and y axis (do not just do log 10 of the values, scale the axis), add a geom smooth that shows the trend of the data, and a dashed line along the diagonal (x=y). If your plot has issues with over-plotting, try to deal with that in a good way.

```
wt1_salmon <- read_tsv("salmon_results/WT1/quant.sf")
```

```
## Rows: 10000 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): Name
## dbl (4): Length, EffectiveLength, TPM, NumReads
##
## 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.
```

```
# Plot the correlation between length and effective length
ggplot(wt1_salmon, mapping = aes(x = Length, y = EffectiveLength)) +
  geom_point(alpha = 0.1) +
  geom_smooth(method = "gam") +
  geom_abline(slope = 1, intercept = 0, colour = "red", linetype = "dashed") +
  scale_x_log10() + scale_y_log10() +
  theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ s(x, bs = "cs")'
```



Comment on on the differences between the trend line and the diagonal line with respect to what you would expect and what you know about effective length - what is the difference between x and y and how do you think they should be correlated?

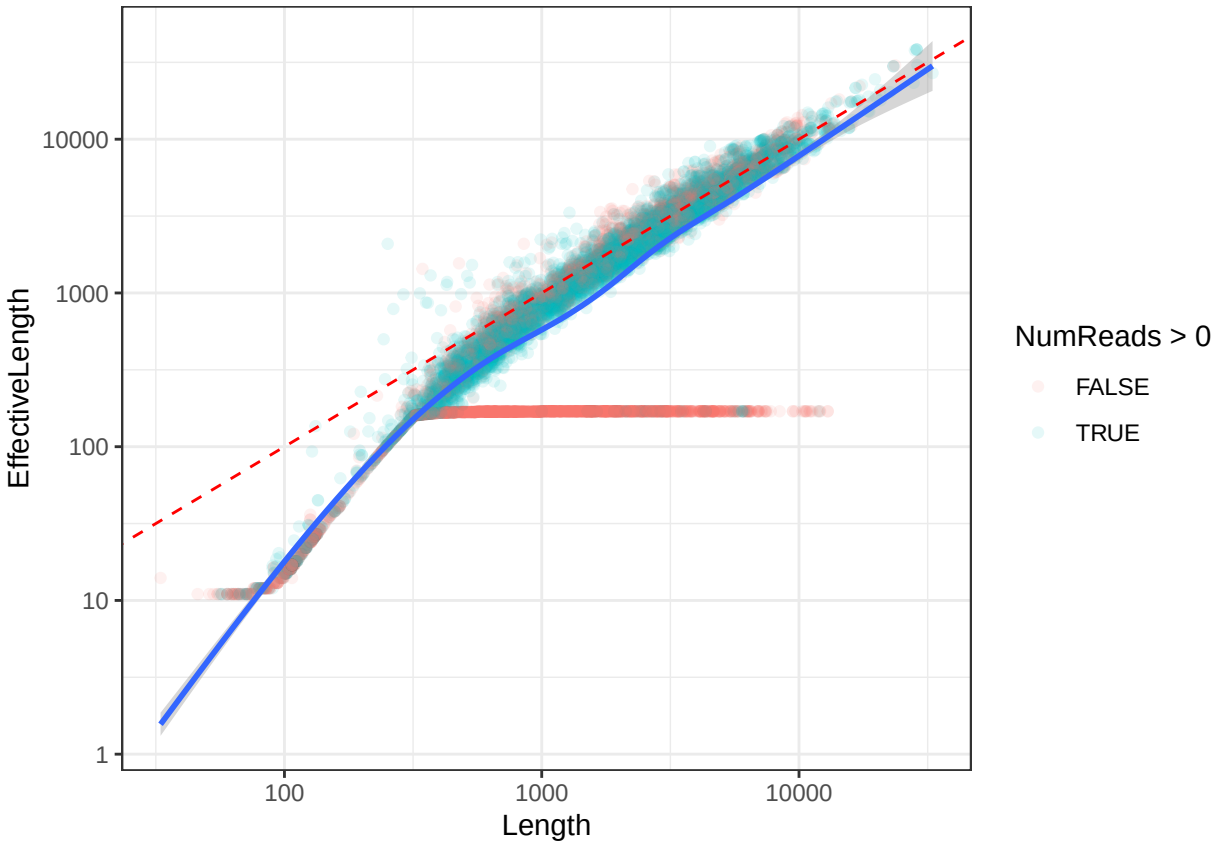
Since the length of the loss region is the same regardless the Length of the transcript, the coverage of the cDNA fragments will be lower for the shorter fragments. In order to reduce the bias, Salmon introduces effective length: $\hat{l_i} = l_i - \mu_d^{l_i}$ where the $\mu_d^{l_i}$ is the mean of truncated fragment length distribution. Thus, we expect linear correlation between the length and effective lenght if there is enough reads to introduce $\mu_d^{l_i}$

After plotting, you may have found a set of outlier points. Explore these points and their properties and see whether you can find what is special with them and whether there is a good way to filter them out that does not rely solely the x and y axis values.

The outliers might be the genes which have very low reads. Based on the Salmon paper, it gives 0 value of effectiveLength if the estimated reads on a transcript is lower than 10^{-8}. Thus we expect that the 0 reads should contribute to the outliers above.

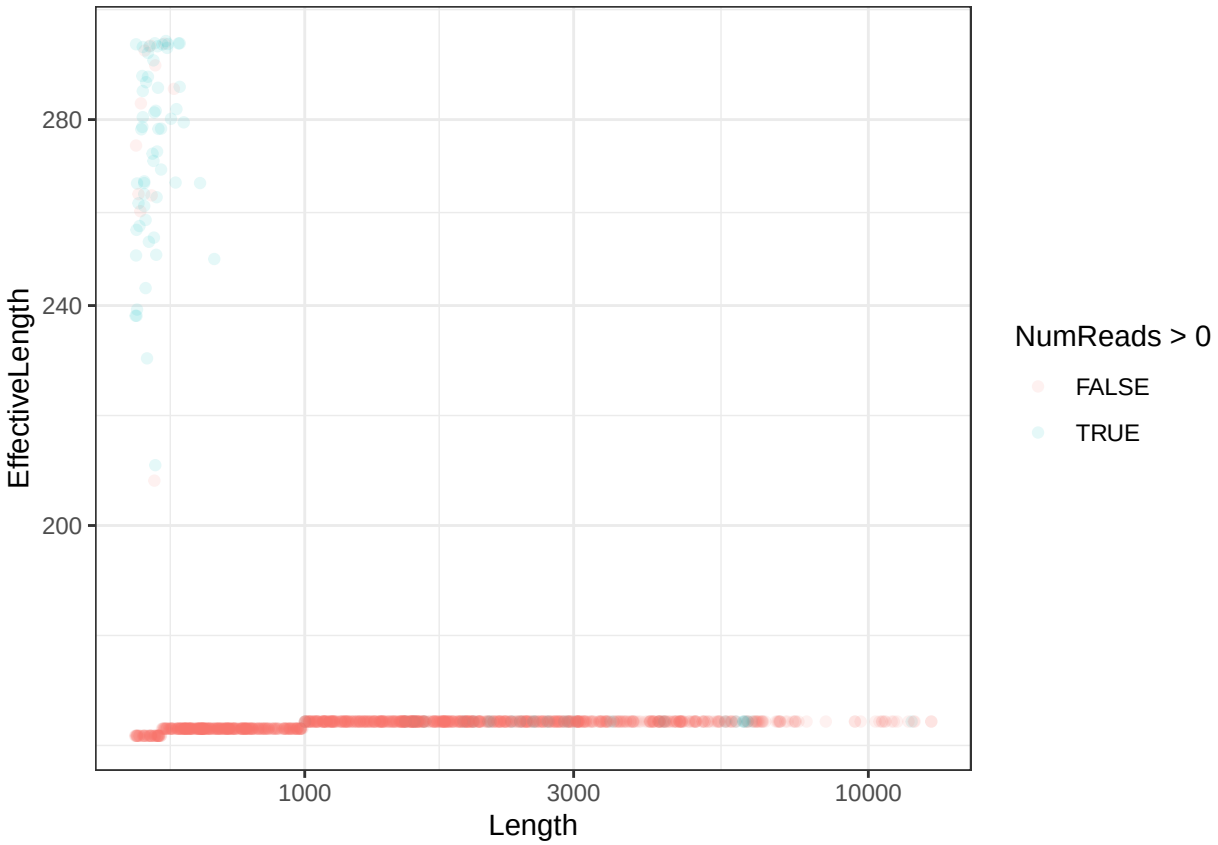
```
# Plot the correlation between length and effective length
ggplot(wt1_salmon, mapping = aes(x = Length, y = EffectiveLength)) +
  geom_point(mapping = aes(colour = NumReads>0), alpha = 0.1) +
  geom_smooth(method = "gam") +
  geom_abline(slope = 1, intercept = 0, colour = "red", linetype = "dashed") +
  scale_x_log10() + scale_y_log10() +
  theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ s(x, bs = "cs")'
```



From the image above we could see that almost all of the outliers are the genes with zero NumReads. However, we could also see that some of the zero reads are still linearly correlated. Thus, it should be noted that this filtering process may have different result for different samples/replicates (e.g. other samples might result in different effective length for a same gene)

```
# Plot the correlation between length and effective length
ggplot(wt1_salmon |> filter(Length>=500 & EffectiveLength<=300),
       mapping = aes(x = Length, y = EffectiveLength)) +
  geom_point(mapping = aes(colour = NumReads>0), alpha = 0.1) +
  scale_x_log10() + scale_y_log10() +
  theme_bw()
```



However, by default, the DESeq library will take care those outliers for us. It uses Cook's distance which measure of how much a single sample is influencing the fitted coefficients for a gene (https://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html#indfilttheory)

Question 2: TPA and Kras KO vs WT

We will now use all samples to explore how cells react to TPA and KRAS KO vs WT. Since both treatments are cancer-inducing, one hypothesis is that the gene expression change vs WT (KO vs WT and TPA vs WT) could be similar.

2.1 : Read in ALL the quant.sf for all WT, WTTPA and KO samples using tximport and make one count and one TPM expression matrix, each of which contains all libraries. Do a 'head´of the abundance matrix.

```
folder_names <- c("KO1", "KO2", "KO3", "WT1", "WT2", "WT3", "WTTPA1",
                  "WTTPA2", "WTTPA3")
path_names <- paste0("./salmon_results/", folder_names, "/quant.sf", sep="") |>
  set_names("KO1", "KO2", "KO3", "WT1", "WT2", "WT3", "WTTPA1",
                  "WTTPA2", "WTTPA3")
file.exists(path_names)
```

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

```
txi <- tximport(path_names, type="salmon", txOut = TRUE)
```

```
## reading in files with read_tsv
```

```
## 1 2 3 4 5 6 7 8 9
```

```
# Abundance equals to TPM,
# counts equals to NumReads,
# Length equals to EffectiveLength

df_count <- as.data.frame(txi$counts)
df_count <- rownames_to_column(df_count, "GeneID")
df_TPM <- as.data.frame(txi$abundance)
df_TPM <- rownames_to_column(df_TPM, "GeneID")

head(df_TPM)
```

```
##           GeneID       KO1       KO2 KO3       WT1      WT2      WT3
## 1 TCONS_00000001 0.0790965 0.1688490   0 0.1853220 0.000000 0.000000
## 2 TCONS_00000002 0.0000000 0.0000000   0 0.0000000 0.000000 0.000000
## 3 TCONS_00000003 0.0000000 0.0000000   0 0.0000000 0.236054 0.134161
## 4 TCONS_00003946 0.0000000 0.0000000   0 0.0223243 0.000000 0.112537
## 5 TCONS_00003947 0.0000000 0.0351832   0 0.0000000 0.427131 0.152389
## 6 TCONS_00003948 0.0000000 0.0000000   0 0.0000000 0.000000 0.000000
##        WTTPA1    WTTPA2 WTTPA3
## 1 2.34738e-01 0.0000000      0
## 2 0.00000e+00 0.0000000      0
## 3 7.87228e-01 0.0000000      0
## 4 0.00000e+00 0.0608736      0
## 5 5.00793e-01 0.0000000      0
## 6 4.38854e-09 0.0000000      0
```

2.2: In the count expression matrix, what fraction rows have exactly zero reads in all samples? How can this be explained?

```
df_count <- df_count |>
  rowwise(GeneID) |>
  mutate(sum_of_reads = sum(c_across(where(is.numeric)))) |>
  ungroup()
print(paste0("Zero reads all samples fraction: ",
             (sum(df_count$sum_of_reads==0.0)/dim(df_count)[1])*100, "%"))
```

```
## [1] "Zero reads all samples fraction: 21.26%"
```

All zeros reads might indicate that those genes are simply not expressed from the mouse skin cells.

2.3: Make an MA plot comparing WT vs WTTPA, and WT vs KO using ggplot (it is fine to use matrix commands also, of course) on the appropriate expression matrix. Do not use any shortcut function that makes an MA plot by itself, e.g. the plotMA function in the deSeq2 package.

```
# Since the result from tximport we don't have to have the EM!
# EM <- df_count |> column_to_rownames(var="GeneID") |> select(!sum_of_reads) |>
#   as.matrix()

ALPHA = 0.05

design <- tibble(Sample = c("KO1", "KO2", "KO3", "WT1", "WT2", "WT3", "WTTPA1",
                  "WTTPA2", "WTTPA3"), Treats = c("KO", "KO", "KO", "WT", "WT",
                                                  "WT", "WTTPA", "WTTPA", "WTTPA"))
## Add pseudocounts to prevent NA values for the LFC
txi$counts <- txi$counts + 1
```

```
dds <- DESeqDataSetFromTximport(txi, colData = design,
                                design = ~Treats)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
## using counts and average transcript lengths from tximport
```

```
res_dds <- DESeq(dds)
```

```
## estimating size factors
```

```
## using 'avgTxLength' from assays(dds), correcting for library size
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
# The low reads on the genes will be filtered out by the results() leaving the
# padj to "NA"
wt_ko <- results(res_dds, contrast = c("Treats", "KO", "WT"),
                 alpha = ALPHA) #default lfcThreshold is zero
tibble_wt_ko <- results(res_dds, contrast =  c("Treats", "KO", "WT"),
                        alpha = ALPHA, tidy = TRUE)
#default lfcThreshold is zero
wt_wttpa <- results(res_dds, contrast = c("Treats", "WTTPA", "WT"),
                 alpha = ALPHA) #default lfcThreshold is zero
tibble_wt_wttpa <- results(res_dds, contrast =  c("Treats", "WTTPA", "WT"),
                        alpha = ALPHA, tidy = TRUE)
#default lfcThreshold is zero
```

```
# Plot MA with GGPLOT
plot_MA <- function(data, title, alpha=0.05) {
  ggplot(data, mapping = aes(x = baseMean, y = log2FoldChange,
                             color=padj < alpha)) +
    geom_point(alpha=0.2) +
    geom_hline(yintercept = 0, alpha = 0.75, color = "red") +
    geom_hline(yintercept = -1) +
    geom_hline(yintercept = 1) +
    geom_smooth(method = "gam", colour = "blue") +
    scale_x_log10() +
    labs(title = title) +
    theme_bw()
}
```

There are three extra requests for each MA plot

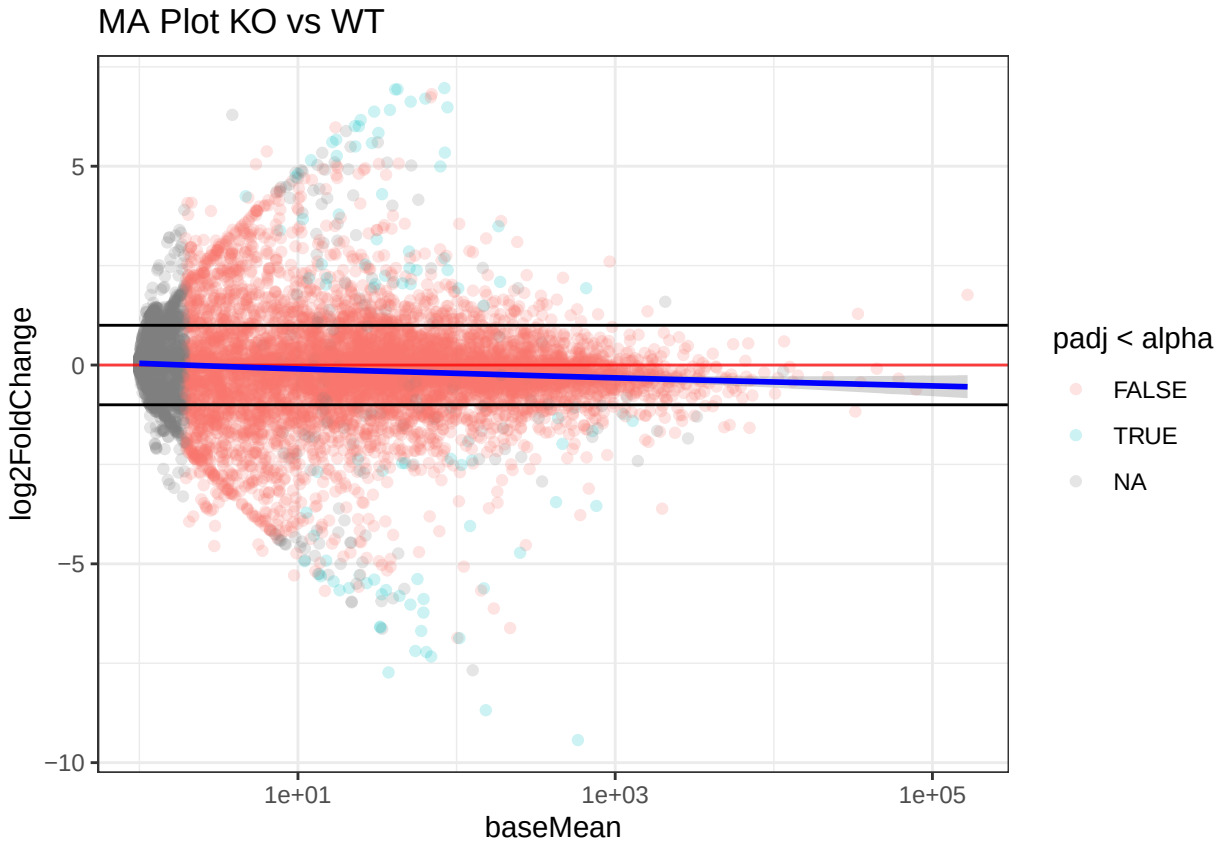
i) If we divide by zero, we will get infinite values, which we know are wrong. To avoid this, we often add a 'pseudo-count' to all values in the matrix - you can view this as adding a small constant noise to all values. In this case, 1 is a good pseudocount value that you should add to all matrix values.

ii) add two black horizontal dashed lines to the plot at log2C 1 and -1, and a red horizontal dashed line at log2C 0.

iii) add a geom_smooth fitted line that follows the data in blue

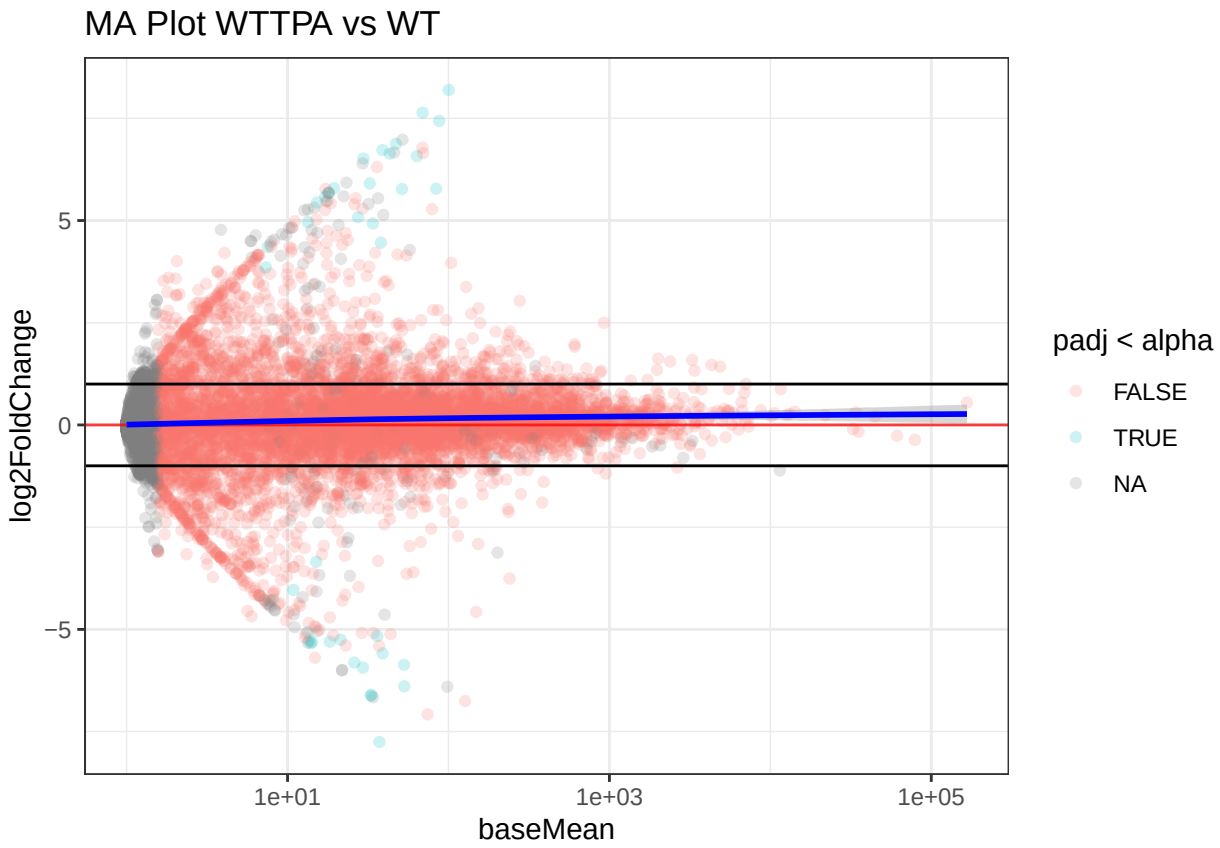
```
plot_MA(tibble_wt_ko, "MA Plot KO vs WT")
```

```
## `geom_smooth()` using formula = 'y ~ s(x, bs = "cs")'
```




```
plot_MA(tibble_wt_wttpa, "MA Plot WTTPA vs WT")
```

```
## `geom_smooth()` using formula = 'y ~ s(x, bs = "cs")'
```

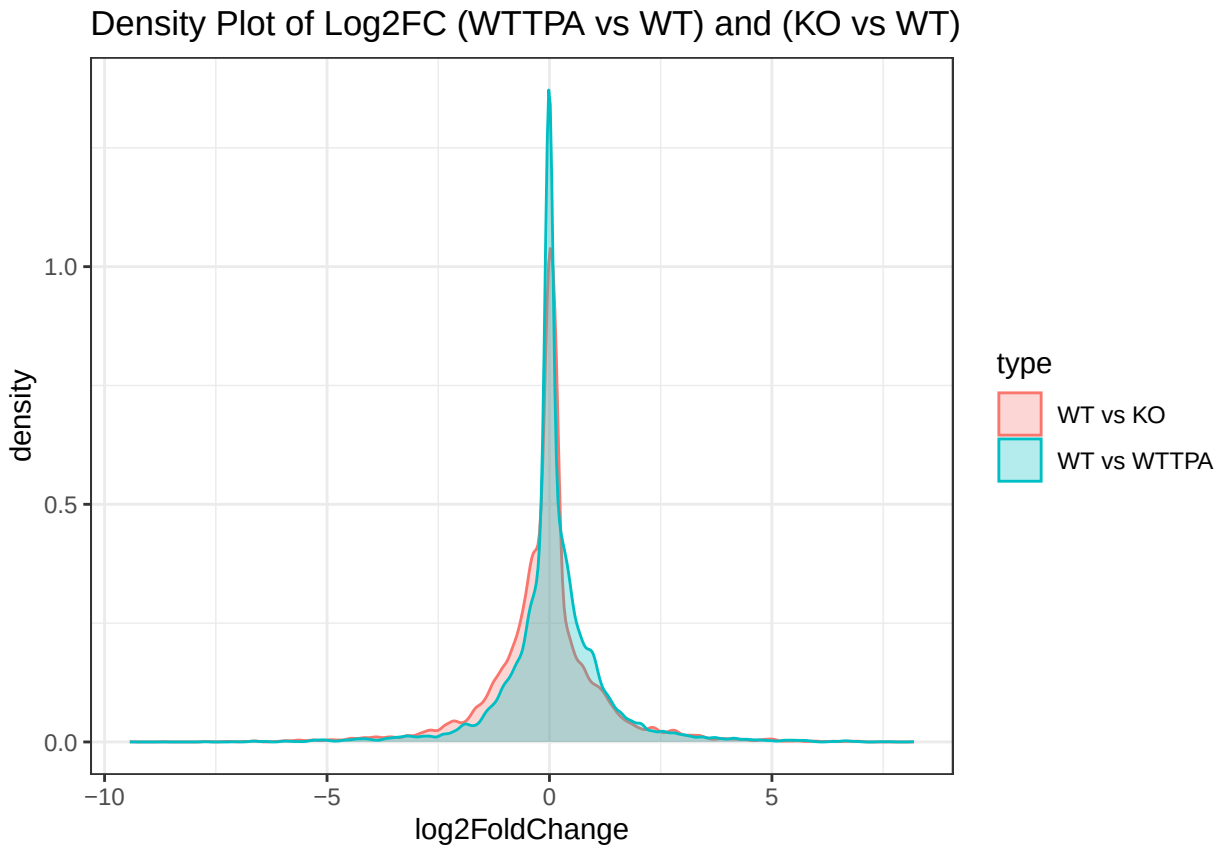


2.4: Interpret the MA plots. What can we see and learn from them? Can we use them to get an idea about the amount of change vs WT for WTTPA and KO, respectively?

From the MA plot we could see, at a glance, whether genes are differentialy expressed between the treatments or not, whether they are up or down-regulated and also their "effect" or the amount of change of each treatments to the genes.

2.5: Continuing from the last question. Make a density plot showing the log2FC distributions (WTTPA vs WT) and (KO vs WT).

```
ggplot(data = bind_rows(tibble_wt_ko |> mutate(type = "WT vs KO"),
                        tibble_wt_wttpa |> mutate(type = "WT vs WTTPA"))) +
  geom_density(mapping = aes(x = log2FoldChange, fill = type, colour = type),
               alpha = 0.3) +
  labs(title = "Density Plot of Log2FC (WTTPA vs WT) and (KO vs WT)") +
  theme_bw()
```



2.6: Make an x-y plot where y is log2FC (WTTPA vs WT) and x is log2FC(KO vs WT).

```
library(ggpmisc)
```

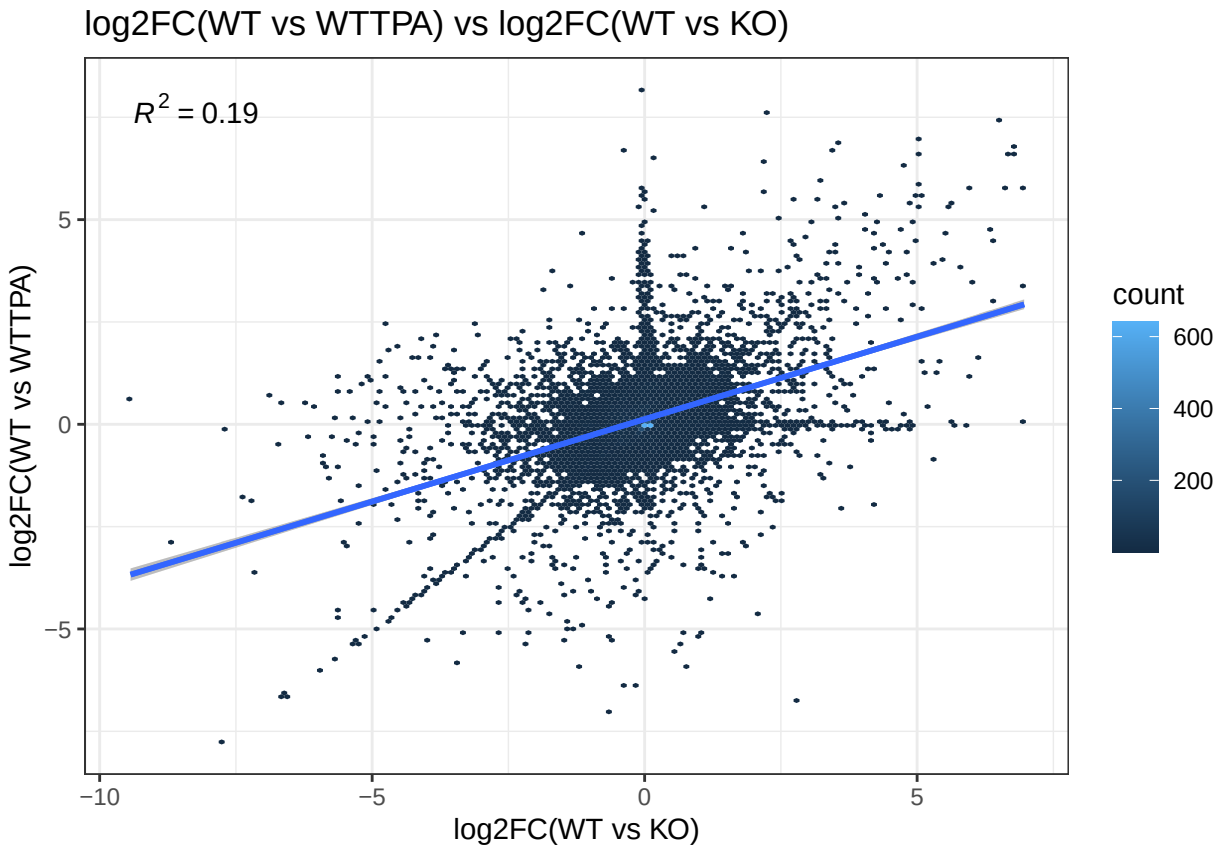
```
## Loading required package: ggpp
```

```
##
## Attaching package: 'ggpp'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
data <- bind_cols(tibble_wt_ko |>
                    select(log2FoldChange) |>
                    rename(log2FoldChange = "log2FC(WT vs KO)"),
                  tibble_wt_wttpa |>
                    select(log2FoldChange) |>
                    rename(log2FoldChange = "log2FC(WT vs WTTPA)"))

ggplot(data = data,
       mapping = aes(x = `log2FC(WT vs KO)`, y = `log2FC(WT vs WTTPA)`)) +
  geom_hex(bins = 150) + stat_poly_line() +
  stat_poly_line() +
  stat_poly_eq() +
  labs(title = "log2FC(WT vs WTTPA) vs log2FC(WT vs KO)") +
  theme_bw()
```



2.7: What do the two above plots show? Interpret the plots and relate them to the hypothesis at the start of question 2.

From the graph above, we expect that the two are linearly correlated (if KO up-regulates particular genes, WTTPA will also up-regulate and vice versa) due to they are cancer-inducing treatments. However, from the graph above, we could see that they are not really correlated with low R^2 values (we could also see from the dispertion of the dots).

We can further do statistic test for it. Since the two seem to be normally distributed based on the density graph and each rows are the same gene, we canpaired t-test. The null hypothesis is log2FC(WT vs KO) and log2FC(WT vs WTTPA) have the same mean (no difference on the effect of the gene) while the alternative is their mean difference is not zero.

```
t.test(data$`log2FC(WT vs KO)`, data$`log2FC(WT vs WTTPA)`, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  data$`log2FC(WT vs KO)` and data$`log2FC(WT vs WTTPA)`
## t = -14.194, df = 9999, p-value < 2.2e-16
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -0.2046876 -0.1550134
## sample estimates:
## mean difference
##      -0.1798505
```

As you can see, the test gives p-value < 2.2e-16 which smaller than our threshold (0.05). Thus, the two have different mean. In conclusion, KO and WTTPA treatments on WT give different effect. It migh be due to

the two treatments effecting different cellular functions on skin cells.