Bioinformatician NGS Technical Interview

Kavya Banerjee

2024-11-02‘

Task #1: Intersecting ChIP-seq peaks with enhancers

You are working in the cell line K562 for which you have a set of annotated enhancers and ChIP-seq peaks for the transcription factors EP300 and CTCF. You are tasked with performing the following comparisons:

Available data: K562 enhancers EP300 ChIP-seq peaks CTCF ChIP-seq peaks

Tasks:

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(purrr)
library(ggplot2)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(stringr)
library(tibble)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(ggrepel)
```

```
## Warning: package 'ggrepel' was built under R version 4.3.3
```

```
library(here)
```

```
## here() starts at /Users/kavyabanerjee/Desktop/bioinformatician_umass_assessment_kavya_banerjee
```

```r
dir_path <- file.path("~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data") # change it as

gz_files <- list.files(path = dir_path, pattern = ".bed.gz$", full.names = TRUE)

# file to unzip
decompress_file <- function(gz_path) {
  command <- paste0("gunzip -dk ", gz_path)
  system(command)
}

purrr::map(gz_files, decompress_file)
```

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] 1
##
## [[3]]
## [1] 1
```

Based on a visual inspection of the bed files, they follow the optional BED extensions. https://genome.ucsc.edu/FAQ/FAQformat.html#format1. A discussion on BED files and indexing https://divingintogeneticsandgenomics.com/post/most-common-mistake-for-bioinformatics/
Here, fetching the bed files:

```r
library(GenomicRanges)
```

```
## Loading required package: stats4

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:tidyr':
##
```

```
##     expand
## The following objects are masked from 'package:dplyr':
##
##     first, rename
## The following object is masked from 'package:utils':
##
##     findMatches
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
## Loading required package: IRanges
##
## Attaching package: 'IRanges'
## The following object is masked from 'package:purrr':
##
##     reduce
## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice
## Loading required package: GenomeInfoDb
## Warning: package 'GenomeInfoDb' was built under R version 4.3.3
```

```
library(rtracklayer)
library(plyranges)
```

```
##
## Attaching package: 'plyranges'
## The following object is masked from 'package:IRanges':
##
##     slice
## The following objects are masked from 'package:dplyr':
##
##     between, n, n_distinct
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.3.3
##
## Attaching package: 'data.table'
## The following object is masked from 'package:plyranges':
##
##     between
## The following object is masked from 'package:GenomicRanges':
##
##     shift
```

```
## The following object is masked from 'package:IRanges':
##
##     shift

## The following objects are masked from 'package:S4Vectors':
##
##     first, second

## The following objects are masked from 'package:reshape2':
##
##     dcast, melt

## The following object is masked from 'package:purrr':
##
##     transpose

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
k562_enhancers_df <- fread(file.path(dir_path, "K562-Enhancers.bed"),
                           select = 1:11,
                              col.names = c("chr", "start", "end", "name", "score",
                                            "strand", "thickStart", "thickEnd",
                                            "itemRgb", "annotation", "classification")) %>%
  mutate(start = start+1)

k562_enhancers <- makeGRangesFromDataFrame(k562_enhancers_df,
                                           seqnames.field = "chr",
                                           start.field = "start",
                                           end.field = "end",
                                           keep.extra.columns = TRUE)

ep300_peaks_df <- fread(file.path(dir_path, "ENCFF433PKW.bed"),
                           select = 1:3,
                              col.names = c("chr", "start", "end")) %>%
  mutate(start = start+1)

ep300_peaks <- makeGRangesFromDataFrame(ep300_peaks_df,
                                        seqnames.field = "chr",
                                        start.field = "start",
                                        end.field = "end")

ctcf_peaks_df <- fread(file.path(dir_path, "ENCFF769AUF.bed"),
                       select = 1:3,
                              col.names = c("chr", "start", "end")
                       ) %>%
  mutate(start = start+1)

ctcf_peaks <- makeGRangesFromDataFrame(ctcf_peaks_df,
                                        seqnames.field = "chr",
                                        start.field = "start",
                                        end.field = "end")
```

1. How many EP300 peaks overlap enhancers? What percentage of EP300 peaks overlap enhancers?

```
peaks_enhancer_overlaps <- findOverlaps(ep300_peaks, k562_enhancers, maxgap = -1L, minoverlap = 0L)
overlapping_peaks <- length(unique(queryHits(peaks_enhancer_overlaps)))
pct_overlapping_peaks <- overlapping_peaks / length(ep300_peaks) * 100

cat("Num. (percentage) of EP300 peaks overlapping enhancers:", overlapping_peaks, "(", pct_overlapping_
```

```
## Num. (percentage) of EP300 peaks overlapping enhancers: 6365 ( 27.26727 %)
```

2. How many enhancers overlap EP300 peaks? What percentage of enhancers overlap EP300 peaks?

```
enhancer_peaks_overlaps <- findOverlaps(k562_enhancers, ep300_peaks, maxgap=-1L, minoverlap=0L,)
overlapping_enhancers <- length(unique(queryHits(enhancer_peaks_overlaps)))
pct_overlapping_enhancers <- overlapping_enhancers / length(k562_enhancers) * 100

cat("Num. (percentage) of K562 enhancers overlapping EP300 peaks:", overlapping_enhancers, "(",  pct_ov
```

```
## Num. (percentage) of K562 enhancers overlapping EP300 peaks: 6732 ( 34.46652 %)
```

3. How many CTCF peaks overlap enhancers? What percentage of CTCF peaks overlap enhancers?

```
ctcf_peaks_enhancer_overlaps <- findOverlaps(ctcf_peaks, k562_enhancers, maxgap = -1L, minoverlap = 0L)
ctcf_overlapping_peaks <- length(unique(queryHits(ctcf_peaks_enhancer_overlaps)))
pct_ctcf_overlapping_peaks <- ctcf_overlapping_peaks / length(ctcf_peaks) * 100

cat("Num. (percentage) of CTCF peaks overlapping enhancers:", ctcf_overlapping_peaks, "(", pct_ctcf_ove
```

```
## Num. (percentage) of CTCF peaks overlapping enhancers: 2391 ( 4.619486 %)
```

4. How many enhancers overlap CTCF peaks? What percentage of enhancers overlap CTCF peaks?

```
enhancer_ctcf_peaks_overlaps <- findOverlaps(k562_enhancers, ctcf_peaks, maxgap = -1L, minoverlap = 0L)
overlapping_enhancers_ctcf <- length(unique(queryHits(enhancer_ctcf_peaks_overlaps)))
pct_overlapping_enhancers_ctcf <- overlapping_enhancers_ctcf / length(k562_enhancers) * 100

cat("Num. (percentage) of K562 enhancers overlapping CTCF peaks:", overlapping_enhancers_ctcf, "(", pct
```

```
## Num. (percentage) of K562 enhancers overlapping CTCF peaks: 2547 ( 13.04014 %)
```

5. Of the enhancers that overlapped the ChIP-seq peaks, how many overlap both EP300 and CTCF? Make a Venn Diagram to illustrate this overlap.

```
ep300_enhancers <- subjectHits(enhancer_peaks_overlaps) %>% unique()
ctcf_enhancers <-  subjectHits(enhancer_ctcf_peaks_overlaps) %>% unique()
ep300_ctcf_enhancers_overlap <- intersect(ep300_enhancers, ctcf_enhancers)
cat("Num. of enhancers overlapping both EP300 and CTCF peaks:", length(ep300_ctcf_enhancers_overlap))
```

```
## Num. of enhancers overlapping both EP300 and CTCF peaks: 268
```

```
library(ggVennDiagram)
```

```
##
## Attaching package: 'ggVennDiagram'
```

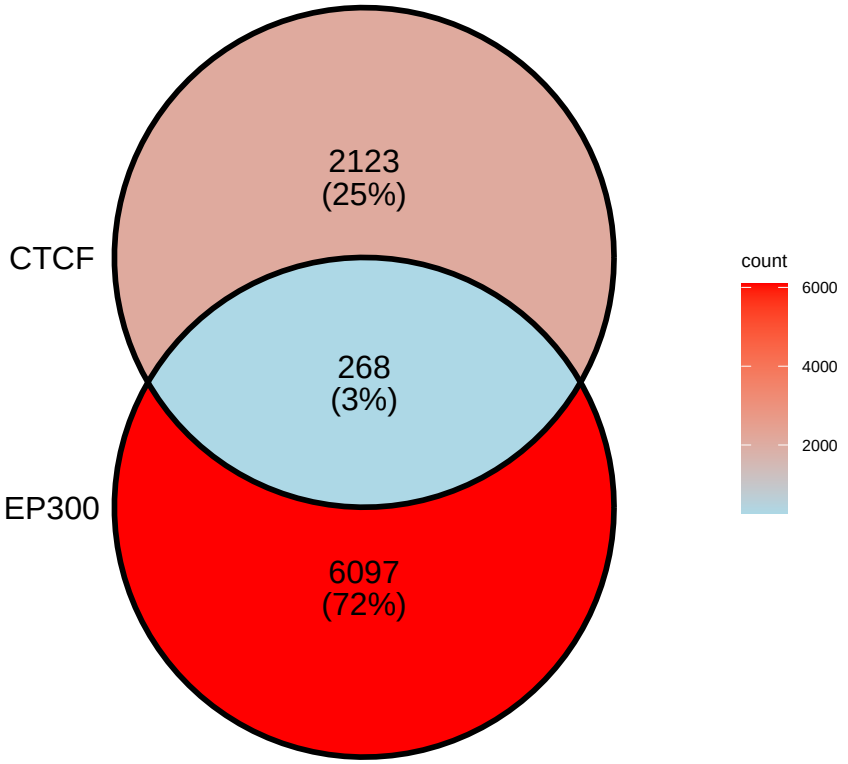
```
## The following object is masked from 'package:tidyr':
##
##     unite
```

```
enhancer_overlap_list <- list(
  `EP300` = ep300_enhancers,
  `CTCF` = ctcf_enhancers
```

```
)

venn_plot <- ggVennDiagram(enhancer_overlap_list, label_alpha = 0) +
  scale_fill_gradient(low = "lightblue", high = "red") +
  labs(title = "Venn Diagram of Enhancer Overlap between EP300 and CTCF") +
  theme(
    plot.title = element_text(size = 10),
    text = element_text(size = 7),
  ) +
  scale_x_continuous(expand = expansion(mult = .2)) +
  scale_color_manual(values = c("black", "black", "black"), guide = FALSE)

# Display the plot
print(venn_plot)
```

```
## Warning: The `guide` argument in `scale_*()` cannot be `FALSE`. This was deprecated in
## ggplot2 3.3.4.
## i Please use "none" instead.
## i The deprecated feature was likely used in the ggVennDiagram package.
##   Please report the issue to the authors.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

Venn Diagram of Enhancer Overlap between EP300 and CTCF



Task 2: Sample swaps

You are working with RNA-seq data from four tissues from two different donors. Your labmate has informed you that the donor labs on two of the samples have been swapped, but does not remember for which tissues. You are tasked with determining which samples have been swapped.

Tasks: 1. Identify which tissues have swapped donors. 2. Attach your code for determining the sample swap. 3. Write a short paragraph detailing how you came to this decision.

```
# creating a sample metadata spec sheet
sample_metadata <- tibble(
    file_name = c("ENCFF338WAN", "ENCFF484GLG", "ENCFF624PWP", "ENCFF253XRU",
                  "ENCFF402FVH", "ENCFF355CVD", "ENCFF649AAK", "ENCFF491GKL"),
    donor = c("Donor_A", "Donor_B", "Donor_A", "Donor_B", "Donor_A", "Donor_B", "Donor_A", "Donor_B"),
    tissue = c("Stomach", "Stomach", "Spleen", "Spleen", "Lung", "Lung", "Colon", "Colon")
) %>%
  mutate(
    file_path = file.path(dir_path, "rnaseq", paste0(file_name, ".tsv")),
    sample_label = paste(file_name)
    ) %>%
  as.data.frame()

rownames(sample_metadata) <- sample_metadata$sample_label

print(sample_metadata)
```

```
##               file_name   donor  tissue
## ENCFF338WAN ENCFF338WAN Donor_A Stomach
## ENCFF484GLG ENCFF484GLG Donor_B Stomach
## ENCFF624PWP ENCFF624PWP Donor_A  Spleen
## ENCFF253XRU ENCFF253XRU Donor_B  Spleen
## ENCFF402FVH ENCFF402FVH Donor_A    Lung
## ENCFF355CVD ENCFF355CVD Donor_B    Lung
## ENCFF649AAK ENCFF649AAK Donor_A   Colon
## ENCFF491GKL ENCFF491GKL Donor_B   Colon
##                                                                                                  file_path
## ENCFF338WAN ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF338WAN.tsv
## ENCFF484GLG ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF484GLG.tsv
## ENCFF624PWP ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF624PWP.tsv
## ENCFF253XRU ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF253XRU.tsv
## ENCFF402FVH ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF402FVH.tsv
## ENCFF355CVD ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF355CVD.tsv
## ENCFF649AAK ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF649AAK.tsv
## ENCFF491GKL ~/Desktop/bioinformatician_umass_assessment_kavya_banerjee/data/rnaseq/ENCFF491GKL.tsv
##             sample_label
## ENCFF338WAN  ENCFF338WAN
## ENCFF484GLG  ENCFF484GLG
## ENCFF624PWP  ENCFF624PWP
## ENCFF253XRU  ENCFF253XRU
## ENCFF402FVH  ENCFF402FVH
## ENCFF355CVD  ENCFF355CVD
## ENCFF649AAK  ENCFF649AAK
## ENCFF491GKL  ENCFF491GKL
```

Counts specifications from here: https://www.encodeproject.org/data-standards/rna-seq/small-rnas/ I'll be using the unstranded counts as the raw counts for normalization

```r
library(readr)

# function to read in files and drop the columns
read_and_process_counts <- function(file_path, file_name, donor, tissue) {
    read_tsv(file_path, col_names = c("gene_id", "unstranded_counts", "read1_stranded_counts", "read2_s
        select(gene_id, unstranded_counts) %>%
        filter(!gene_id %in% c("N_unmapped", "N_multimapping", "N_noFeature", "N_ambiguous")) %>%
        # dplyr::rename(!!paste(file_name, donor, tissue, sep = "_") := unstranded_counts)
    dplyr::rename(!!paste(file_name) := unstranded_counts)
}

# call the function for the multiple samples
quantification_list <- pmap(
    list(sample_metadata$file_path, sample_metadata$file_name, sample_metadata$donor, sample_metadata$ti
    read_and_process_counts
)
```

```
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
##
## 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.
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
##
## 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.
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
##
## 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.
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
##
## 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.
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
```

```
##
## 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.
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
##
## 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.
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
##
## 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.
## Rows: 61378 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): gene_id
## dbl (3): unstranded_counts, read1_stranded_counts, read2_stranded_counts
##
## 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.
```

```r
quantification_df <- purrr::reduce(quantification_list, full_join, by = "gene_id")

head(quantification_df)
```

```
## # A tibble: 6 x 9
##   gene_id            ENCFF338WAN ENCFF484GLG ENCFF624PWP ENCFF253XRU ENCFF402FVH
##   <chr>                    <dbl>       <dbl>       <dbl>       <dbl>       <dbl>
## 1 ENSG00000223972.5            0           0           0           0           0
## 2 ENSG00000227232.5            0           0           0           0           0
## 3 ENSG00000278267.1            0           0           0           0           0
## 4 ENSG00000243485.3            0           0           0           0           0
## 5 ENSG00000274890.1            0           0           0           0           0
## 6 ENSG00000237613.2            0           0           0           0           0
## # i 3 more variables: ENCFF355CVD <dbl>, ENCFF649AAK <dbl>, ENCFF491GKL <dbl>
```

```r
count_matrix <- quantification_df %>%
    column_to_rownames(var = "gene_id") %>%
    as.matrix()

dim(count_matrix)
```

```
## [1] 61374     8
```

```r
library(DESeq2)
```

```
## Warning: package 'DESeq2' was built under R version 4.3.3
```

```
## Loading required package: SummarizedExperiment
```

```
## Loading required package: MatrixGenerics

## Loading required package: matrixStats

## Warning: package 'matrixStats' was built under R version 4.3.3

##
## Attaching package: 'matrixStats'

## The following object is masked from 'package:dplyr':
##
##     count

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

```
count_matrix <- round(count_matrix)

# normalize via deseq2 - just account for technical variation and tissue, since i don't want to bias th
dds <- DESeqDataSetFromMatrix(
    countData = count_matrix,
    colData = sample_metadata,
    # design = ~ 1
```

```
    design = ~ tissue
)
```

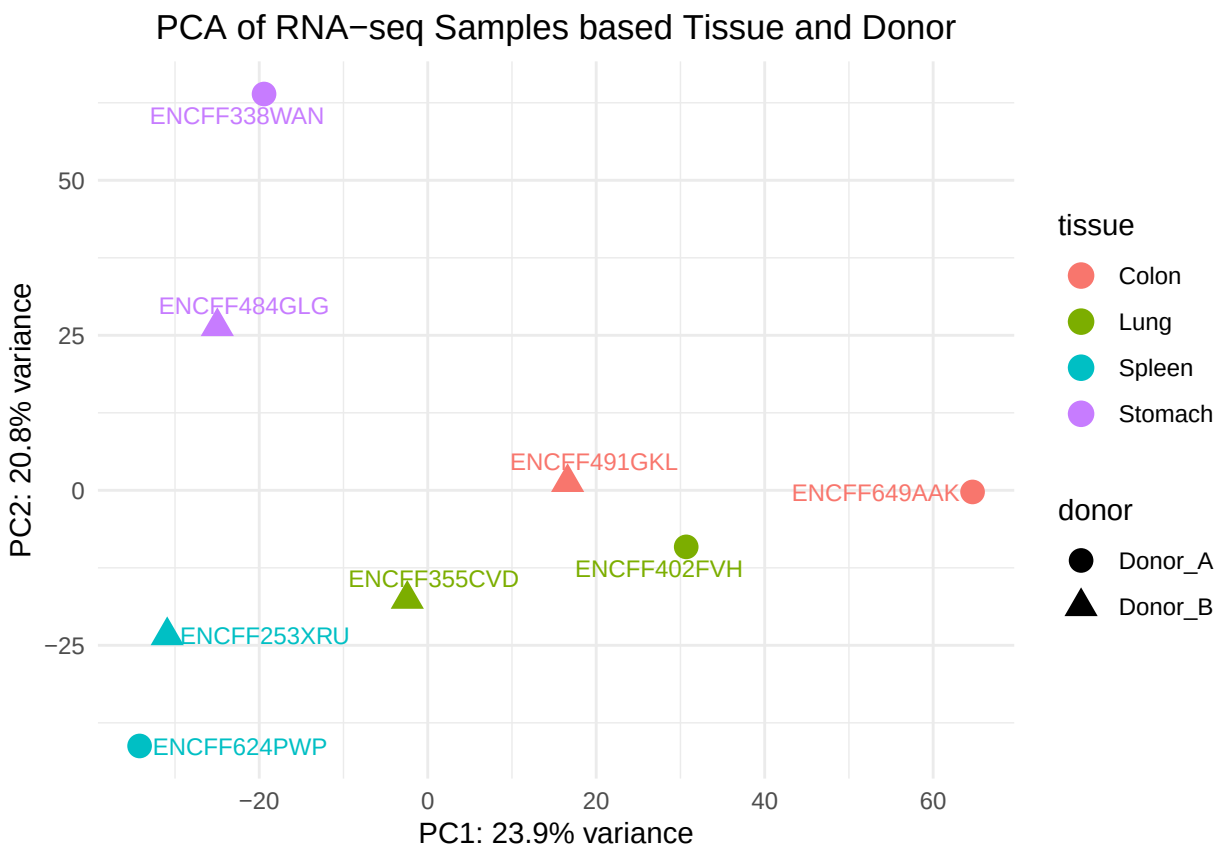
```
## converting counts to integer mode
```

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

```
# filter low counts overall
smallestGroupSize <- 3
keep <- rowSums(counts(dds) >= 10) >= smallestGroupSize
dds <- dds[keep,]

vsd <- vst(dds, blind=TRUE) # for no bias in design
vsd_mat <- assay(vsd)
dim(vsd_mat)
```

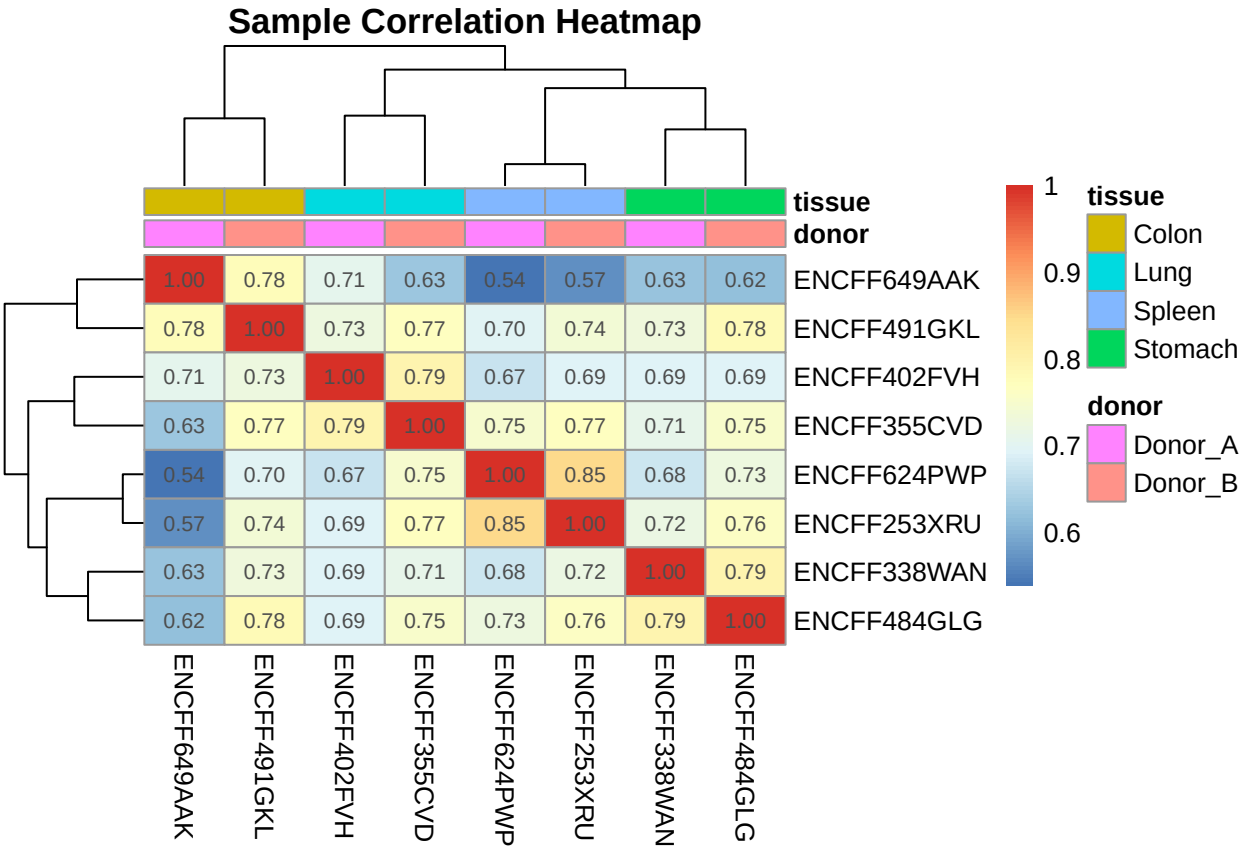
```
## [1] 5107    8
```

```
pca <- prcomp(t(vsd_mat), scale. = TRUE)

percentVar <- pca$sdev^2 / sum(pca$sdev^2) * 100

#
pca_df <- as.data.frame(pca$x) %>%
    rownames_to_column(var = "sample") %>%
    left_join(sample_metadata, by = c("sample" = "sample_label"))

# plot with tissue and donor
ggplot(pca_df, aes(x = PC1, y = PC2, color = tissue, shape = donor)) +
    geom_point(size=4) +
    geom_text_repel(aes(label=sample), size=3, max.overlaps = 10) +
    theme_minimal() +
    labs(title = "PCA of RNA-seq Samples based Tissue and Donor",
         x = paste0("PC1: ", round(percentVar[1], 1), "% variance"),
         y = paste0("PC2: ", round(percentVar[2], 1), "% variance")) +
    theme(plot.title = element_text(hjust = 0.5))
```



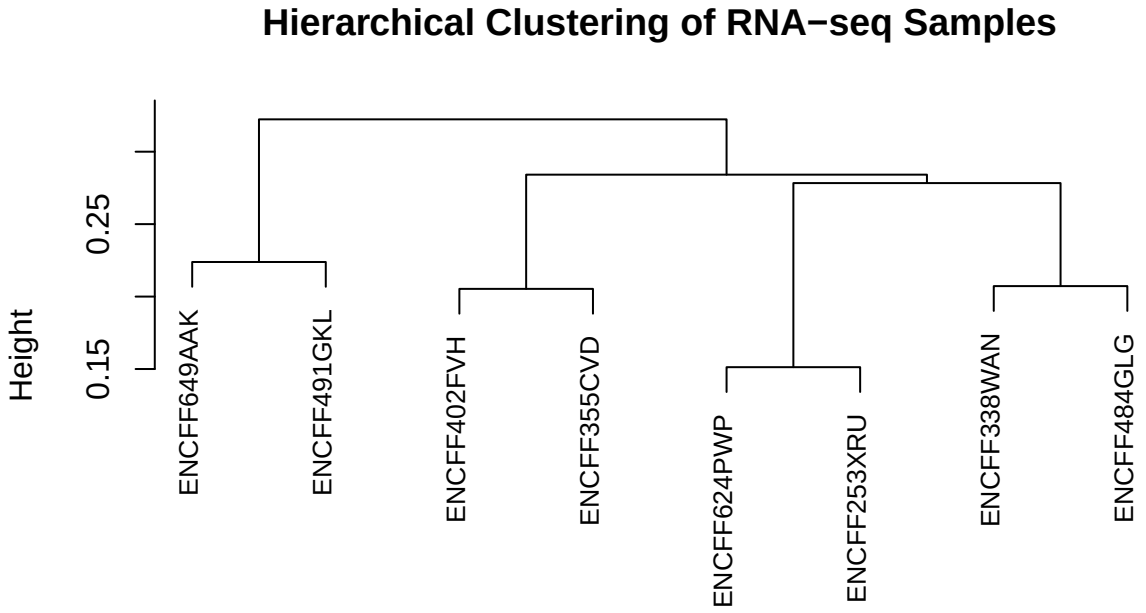
In the PCA plot, we see an interesting trend: samples from Donor A tend to cluster higher along PC2 than those from Donor B across most tissues—like the stomach, lung, and colon, hints at a stable, donor-specific expression pattern across these tissues. However, the spleen samples break this pattern; Donor B's spleen sample actually clusters higher on PC2 than Donor A's, flipping the usual trend. This unexpected result raises a question—there might be a labeling issue with the spleen samples, given the deviation from the expected donor-specific clustering. PC1 is likely tissue-specific clustering and PC2 donor-specific.

```
library(pheatmap)

# get correlation as heatmap
cor_matrix <- cor(vsd_mat)

rownames(sample_metadata) <- NULL

# correlation
annotation_col <- sample_metadata %>%
    column_to_rownames(var = "sample_label") %>%
    select(donor, tissue)

pheatmap(cor_matrix,
         annotation_col = annotation_col,
         main = "Sample Correlation Heatmap",
         display_numbers = TRUE,
         fontsize_number = 8,
         clustering_distance_rows = "correlation",
         clustering_distance_cols = "correlation",
         clustering_method = "average")
```



These correlation values are relatively similar across all tissues. Interestingly, the correlation between the spleen and colon samples from Donor A and Donor B is slightly high—mirroring (0.85) the kind of correlation pattern we'd expect to see between tissues from the same donor rather than across donors. On top of that, these samples show lower correlations with other tissues from their labeled donor, hinting that they might actually belong to the opposite donor. Given the PCA observations, spleen can be the suspicious ones but it's difficult to say.

```
hc <- hclust(as.dist(1 - cor_matrix), method = "average") # inconclusive

plot(hc, main = "Hierarchical Clustering of RNA-seq Samples", xlab = "", sub = "", labels = rownames(co
```



As expected, samples predominantly clustered based on tissue type, indicating that tissue-specific expression profiles are the main drivers of variation.

```
correlation_df <- as.table(cor_matrix) %>% as.data.frame()
colnames(correlation_df) <- c("Sample1", "Sample2", "Correlation")

correlation_df <- correlation_df %>%
  left_join(sample_metadata %>% select(sample_label, donor, tissue), by = c("Sample1" = "sample_label")) %>%
  left_join(sample_metadata %>% select(sample_label, donor, tissue), by = c("Sample2" = "sample_label"))

# correlations between different donors (cross-donor, same-tissue)
cross_donor_correlations <- correlation_df %>%
  rowwise() %>%
  mutate(
    Sample1 = ifelse(donor_1 == "Donor_B", Sample2, Sample1),
    Sample2 = ifelse(donor_1 == "Donor_B", Sample1, Sample2),
    donor_1 = ifelse(donor_1 == "Donor_B", donor_2, donor_1),
    donor_2 = ifelse(donor_1 == "Donor_B", donor_1, donor_2),
    tissue_1 = ifelse(donor_1 == "Donor_B", tissue_2, tissue_1),
    tissue_2 = ifelse(donor_1 == "Donor_B", tissue_1, tissue_2)
  ) %>%
  ungroup() %>%
  filter(donor_1 != donor_2 & tissue_1 == tissue_2) %>%
  distinct(Sample1, Sample2, .keep_all = TRUE)

print("Cross-donor correlations within the same tissue:")
```

```
## [1] "Cross-donor correlations within the same tissue:"
```

```
print(cross_donor_correlations)
```

```
## # A tibble: 4 x 7
##   Sample1     Sample2     Correlation donor_1 tissue_1 donor_2 tissue_2
##   <chr>       <chr>             <dbl> <chr>   <chr>    <chr>   <chr>
## 1 ENCFF338WAN ENCFF484GLG       0.793 Donor_A Stomach  Donor_B Stomach
## 2 ENCFF624PWP ENCFF253XRU       0.849 Donor_A Spleen   Donor_B Spleen
```

```
## 3 ENCFF402FVH ENCFF355CVD       0.795 Donor_A Lung     Donor_B Lung
## 4 ENCFF649AAK ENCFF491GKL       0.776 Donor_A Colon    Donor_B Colon
```

```
# correlations between different tissues within same donor (within-donor, cross-tissue)
within_donor_correlations <- correlation_df %>%
  filter(donor_1 == donor_2 & tissue_1 != tissue_2) %>%
  rowwise() %>%
  mutate(
    tissue_pair = paste(sort(c(tissue_1, tissue_2)), collapse = "_")
  ) %>%
  ungroup() %>%
  distinct(donor_1, tissue_pair, .keep_all = TRUE) %>%
  arrange(donor_1, donor_2) %>%
  select(-tissue_pair)

print("Within-donor correlations across different tissues:")
```

```
## [1] "Within-donor correlations across different tissues:"
```

```
print(within_donor_correlations)
```

```
## # A tibble: 12 x 7
##    Sample1     Sample2     Correlation donor_1 tissue_1 donor_2 tissue_2
##    <chr>       <chr>             <dbl> <chr>   <chr>    <chr>   <chr>
##  1 ENCFF624PWP ENCFF338WAN       0.675 Donor_A Spleen   Donor_A Stomach
##  2 ENCFF402FVH ENCFF338WAN       0.691 Donor_A Lung     Donor_A Stomach
##  3 ENCFF649AAK ENCFF338WAN       0.626 Donor_A Colon    Donor_A Stomach
##  4 ENCFF402FVH ENCFF624PWP       0.672 Donor_A Lung     Donor_A Spleen
##  5 ENCFF649AAK ENCFF624PWP       0.539 Donor_A Colon    Donor_A Spleen
##  6 ENCFF649AAK ENCFF402FVH       0.707 Donor_A Colon    Donor_A Lung
##  7 ENCFF253XRU ENCFF484GLG       0.762 Donor_B Spleen   Donor_B Stomach
##  8 ENCFF355CVD ENCFF484GLG       0.753 Donor_B Lung     Donor_B Stomach
##  9 ENCFF491GKL ENCFF484GLG       0.779 Donor_B Colon    Donor_B Stomach
## 10 ENCFF355CVD ENCFF253XRU       0.766 Donor_B Lung     Donor_B Spleen
## 11 ENCFF491GKL ENCFF253XRU       0.741 Donor_B Colon    Donor_B Spleen
## 12 ENCFF491GKL ENCFF355CVD       0.769 Donor_B Colon    Donor_B Lung
```

```
cross_donor_avg <- cross_donor_correlations %>%
  group_by(tissue_1) %>%
  summarise(avg_cross_donor_correlation = mean(Correlation))

cat("Average cross-donor correlations within the same tissue:")
```

```
## Average cross-donor correlations within the same tissue:
```

```
cat(capture.output(glimpse(cross_donor_avg)), sep = "\n")
```

```
## Rows: 4
## Columns: 2
## $ tissue_1                    <chr> "Colon", "Lung", "Spleen", "Stomach"
## $ avg_cross_donor_correlation <dbl> 0.7761310, 0.7946917, 0.8487336, 0.7927786
```

```
within_donor_avg <- within_donor_correlations %>%
  group_by(donor_1) %>%
  summarise(avg_within_donor_correlation = mean(Correlation))

cat("Average within-donor correlations across different tissues:")
```

```
## Average within-donor correlations across different tissues:
```

```
cat(capture.output(glimpse(within_donor_avg)), sep = "\n")
```

```
## Rows: 2
## Columns: 2
## $ donor_1                       <chr> "Donor_A", "Donor_B"
## $ avg_within_donor_correlation <dbl> 0.6516279, 0.7614572
```

Spleen's cross-donor correlation (0.84) is higher than other tissues and even exceeds the within-donor, cross-tissue correlations. Typically expect cross-donor correlations within the same tissue to be lower than within-donor correlations across tissues due to donor-specific gene expression differences. It's likely that the spleen samples are suspicious sample swaps, but a more robust comparison would be use the BAM files for the spleen samples and compare with other samples like via Picard's CrosscheckFingerprints tools (couldn't verify due to memory limitations within my system) .

```
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Sonoma 14.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] pheatmap_1.0.12             DESeq2_1.42.1
##  [3] SummarizedExperiment_1.32.0 Biobase_2.62.0
##  [5] MatrixGenerics_1.14.0       matrixStats_1.4.1
##  [7] readr_2.1.5                 ggVennDiagram_1.5.2
##  [9] data.table_1.16.2           plyranges_1.22.0
## [11] rtracklayer_1.62.0          GenomicRanges_1.54.1
## [13] GenomeInfoDb_1.38.8         IRanges_2.36.0
## [15] S4Vectors_0.40.2            BiocGenerics_0.48.1
## [17] here_1.0.1                  ggrepel_0.9.6
## [19] viridis_0.6.5               viridisLite_0.4.2
## [21] tibble_3.2.1                stringr_1.5.1
## [23] reshape2_1.4.4              ggplot2_3.5.1
## [25] purrr_1.0.2                 tidyr_1.3.1
## [27] dplyr_1.1.4
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1         farver_2.1.2             Biostrings_2.70.3
##  [4] bitops_1.0-9             fastmap_1.2.0            RCurl_1.98-1.16
##  [7] GenomicAlignments_1.38.2 XML_3.99-0.17            digest_0.6.37
```

```
## [10] lifecycle_1.0.4          magrittr_2.0.3           compiler_4.3.2
## [13] rlang_1.1.4              tools_4.3.2              utf8_1.2.4
## [16] yaml_2.3.10              knitr_1.48               S4Arrays_1.2.1
## [19] labeling_0.4.3           bit_4.5.0                DelayedArray_0.28.0
## [22] RColorBrewer_1.1-3       plyr_1.8.9               abind_1.4-8
## [25] BiocParallel_1.36.0      withr_3.0.2              grid_4.3.2
## [28] fansi_1.0.6              colorspace_2.1-1         scales_1.3.0
## [31] tinytex_0.54             cli_3.6.3                rmarkdown_2.28
## [34] crayon_1.5.3             generics_0.1.3           rstudioapi_0.17.1
## [37] tzdb_0.4.0               rjson_0.2.23             zlibbioc_1.48.2
## [40] parallel_4.3.2           XVector_0.42.0           restfulr_0.0.15
## [43] vctrs_0.6.5              Matrix_1.6-5             hms_1.1.3
## [46] bit64_4.5.2              locfit_1.5-9.10          glue_1.8.0
## [49] codetools_0.2-20         stringi_1.8.4            gtable_0.3.6
## [52] BiocIO_1.12.0            munsell_0.5.1            pillar_1.9.0
## [55] htmltools_0.5.8.1        GenomeInfoDbData_1.2.11  R6_2.5.1
## [58] rprojroot_2.0.4          vroom_1.6.5              evaluate_1.0.1
## [61] lattice_0.22-6           highr_0.11               Rsamtools_2.18.0
## [64] Rcpp_1.0.13-1            gridExtra_2.3            SparseArray_1.2.4
## [67] xfun_0.49                pkgconfig_2.0.3
```