Omics analysis on the effect of SARS-CoV-2 in human patients and human cell lines

Yunseol Park and Emin Araftpoor
Supervisor: Prof. dr. ir. Tim De Meyer
Counselors: Louis Coussement, Menno Van Damme

Abstract. SARS-CoV-2 infection is the cause of the worldwide pandemic, COVID-19. Its symptoms range from asymptotic to inflammatory responses to even death. In this report, we collected different omics data on SARS-CoV-2 infection in human patients and cell lines. Multi-omics data analyses can aid in elucidating the pathology of the disease. As such, we made use of a genomic dataset, two transcriptomic datasets, and an epigenomic dataset to grant insight into the mechanisms underlying SARS-CoV-2 infection in human hosts.

Through these analyses, we found that there are significant differences between healthy donors and COVID-19 patients, mostly concerning genes involved in the immune response, cell cycle, and protein production and processing. A significant gene found commonly in three of the four analyses is *DUSP1*. This gene is involved in the immune response and the *MAPK signalling pathway*, and has already been described within the context of SARS-CoV-2 pathology previously (Goel et al., 2021).

Keywords— ChIP-seq; COVID-19; high throughput transcription profiling; methylation profiling; multi-omics data analysis; RNA-seq; SARS-CoV-2.

I. Introduction

SARS-CoV-2, a type of coronavirus, is the cause of the pandemic which has held the world in its grip for over two years. It has cumulatively caused over 643 million cases and over 6 million deaths since its outbreak (WHO, n.d.). While most infections may be asymptotic or cause mild symptoms, some may cause severe inflammatory responses and even death (Barturen et al., 2022).

Due to the severity and the global scale of the disease, a lot of research efforts were put into unveiling the mechanisms of the SARS-CoV-2 virus and its interaction with human hosts. Infection of SARS-CoV-2 may cause changes in the hosts' genome, such as gene expression, methylation, histone modification, and others (Galván-Peña et al., 2021; Brunetta et al., 2021; Kee et al., 2022; Barturen et al., 2022). Here, we investigate the effect of the infection on the host genome using a multi-omics approach.

II. Datasets and Quality Control

In this multi-omics analysis, Infinium data, RNA-seq data of T-cells, RNA-seq data of monocytes, and ChIP-seq data were used. The overview of all data used in this study can be found in Table 1. All quality control and preprocessing were performed in this study with the use of raw datasets.

1. Infinium Data

Whole blood samples were collected from patients and subjected to methylation profiling using Illumina's Infinium MethylationEPIC BeadChip. Patients were tested for SARS-CoV-2, and in case of a positive result, they were separated into "severe" and "mild" groups based on the WHO clinical ordinal scale (Barturen et al., 2022).

Out of the total 500 samples, only 10 samples were taken. 5 samples from the "severe" group and 5 samples from the "negative" group were randomly chosen for analysis. A small number of samples were chosen for the analysis due to the computational restraints. Previously, 5 additional samples of the "mild" group were also chosen but removed as the paired t-test did not show significant results between the "negative" group and the "mild" group (data not shown).

2. CD4+ Treg Cell RNA-seq Data

Peripheral blood samples were taken from healthy donors and COVID-19 patients, divided into three groups, "mild" for outpatients. "severe" if hospitalized, and "recovered" patients. Peripheral blood mononuclear cells (PBMCs) were isolated, from which the Treg and Tconv cells were isolated via magnetic isolation. Samples with purity above 65% were selected and sequenced via NextSeq 500 to generate 38bp paired-end reads per sample (Galván-Peña et al., 2021).

Out of 86 samples, only 12 samples were selected, 6 healthy donor samples, and 6 COVID-19 patients

Table 1: Data overview. Columns Methylation, T-cell, Monocytes, and ChIP-seq refer to Infinium data, RNA-seq of Treg cells, RNA-seq of monocytes, and ChIP-seq data, respectively. Rows # Original and # Analysed refer to the number of samples in the original dataset and the number of samples analysed.

	Methylation	T-cell	Monocytes	ChIP-seq
Accession	GSE179325	GSE179448	GSE160351	GSE205369
# Original	500	86	9	22
# Analysed	10	12	6	10
Healthy	5	6	3	6
Infected	5	6	3	4
Read length	NA	38 (x2)	75 (x2)	42 (x2)
Single/paired	NA	Paired	Paired	Paired
Platform	Infinium Epic	NextSeq 500	NextSeq 550	NextSeq 500
Source	Whole blood	CD4+ Treg cells	Monocytes	A549 cell lines

in the "severe" group. Furthermore, only CD4+ Treg cells were selected, as Treg cells are more involved in the immediate immune response. More samples were planned to be added to the analysis after the initial 12, but due to the HPC maintenance, we only made use of the 12 samples.

The QC test of the data showed that the variation in the number of reads between samples is large, ranging from 1 to 7 million reads. Furthermore, it shows that the sequencing depth is also quite low, with the total number of reads not even being 10 million. The samples also contain a lot of overrepresented sequences, and these are assumed to be due to either the ribosomal RNA (rRNA) or custom adapter sequences (see Appendix B.1).

The data were trimmed by quality and no adapters were trimmed as no adapter sequences were reported in the initial FastQC (Andrews, 2010), and while overrepresented sequences were abundant, we did not remove them in case these they are significant sequences. If they are indeed rRNA or adapter sequences, they would be removed during the alignment process.

3. Monocyte RNA-seq Data

Peripheral blood samples were taken from healthy donors and COVID-19 patients, from which, PBMCs were isolated. Monocytes were filtered from the PBMC samples and samples with a purity of monocytes above 89% were selected. These monocyte samples were then sequenced via NextSeq 550 and were reported to generate "at least 80 million 75-bp paired-end reads per sample" (Brunetta et al., 2021).

Only 6 samples, 3 healthy donor samples, and 3 COVID-19 patient samples, from the 9 were taken for the data analysis due to the failed QC of the other three samples. The three samples had distinct GC-content trends that diverged from the rest. Some of the remaining samples, especially from the COVID-19 patients, still follow a trend that diverges from rest but are not as distinctive as the removed samples. Moreover, the slight divergence of the GC-content trend was not significant enough to completely fail the QC test.

Furthermore, the QC test showed that the maximum number of reads was not 80 million as reported by Brunetta et al. (2021), but around 16 million before trimming. The number of reads also showed large variability between samples, ranging from 6 to 16 million reads (see Appendix C.1).

The data were trimmed by quality. No adapters were trimmed as no adapter sequences in the initial FastQC. Overrepresented sequences were not removed, following the same reasoning as with the Treg cells.

4. ChIP-seq Data

$A549^{ACE}$ cells, A459 cells expressing the ACE2 receptor, were grown and mock-infected or infected with wild-type SARS-CoV-2 virus. After 48 hours of infection, the cells were lysed to isolate DNA, and subjected to ChIP-seq, immunoprecipitating antibodies specific for the H3K9ac histone modification. The resultant DNA was then sequenced by NextSeq 500 to obtain 42-bp paired reads per sample (Kee et al., 2022).

Only 5 samples (with +5 input controls) were used from 11 (+11 input controls). This was because the other samples were infected with mutated SARS-CoV-2, which was not of interest in this study. The 5 samples consist of 3 replicates of mock infection and 2 replicates of SARS-CoV-2 infection. One replicate of SARS-CoV-2, replicate 2, was originally missing from the dataset and it was not mentioned why this replicate was omitted.

The QC test showed that the number of reads varied in large amounts between samples. The maximum number of reads was above 17 million reads while the minimum was less than 3 million reads after trimming. Even after peak calling, a lot of variation in the number of reads can still be observed. However, when the replicates are pooled together into one peak, it seems

to have slightly less variation. Furthermore, the GC-distribution graph showed one sample as having slight variation from the trend of the other samples. However, this variation was not large enough to fail the test completely, and as the dataset only had a few samples, the sample was not removed (see Appendix D.1).

The data were trimmed by quality. No adapter or overexpressed sequences were reported in the initial FastQC, as such these were not trimmed or removed.

III. Methods

For all the downstream analysis methods, R 4.2.1 was used, and the session info can be found in Appendix E.

1. Infinium Analysis

The methylation degree of about 850 thousand CpG's was measured for each sample. A comparison was then drawn between 5 patients afflicted with a severe SARS-CoV-2 infection and 5 negative, healthy donors.

Probes that generated *NA* values for some samples and those for which the methylation degree was uncertain (*P-value* >0.05) were removed. Data were normalized using the *wateRmelon* package. Significant differential methylation was determined using the *Limma* workflow. This was done with a linear model, taking disease state into account, and adding age, and gender effects as confounders. The results were adjusted for multiple testing using the *Benjamini-Hochberg* procedure.

The details of the analysis in R markdown can be found in Appendix A.2.

2. CD4+ Treg Cell Analysis

Using the selected data, alignment with Kallisto (Bray et al., 2016) was performed. We then performed differential expression (DE) analysis using the *EdgeR* workflow. The library sizes were normalized, and genes were filtered out based on counts. Here, almost twice more genes were filtered out than kept. Then the design matrix and contrast were generated, with the condition of the patients, healthy or COVID-19 infected. The data most likely contains other confounders and factors that may affect the analysis, but as the patient information was not reported, we were not able to include them in the design matrix. After that, the dispersion was calculated and the quasi-likelihood test was performed. Finally, the results were adjusted for multiple testing with the *Benjamini-Hochberg* method to obtain *FDR* values.

Additionally, gene set analysis (GSA), specifically overrepresentation analysis (ORA) using the *WebGestaltR* package, was performed to gain more insight into the types of genes from the DE result and to compare our results to others. Two different analyses were performed, one with the KEGG pathway and the other with a custom gene set. For both analyses, the default *FDR* threshold of 0.05 was used.

For the custom gene set, a gene set consisting of up- or down-regulated genes from blood samples of COVID-19 patients was used. This was obtained from both the literature (Daamen et al., 2021) and "The COVID-19 Drug and Gene Set Library" (Kuleshov et al., 2020). "The COVID-19 Drug and Gene Set Library" is a gene set library containing COVID-related gene sets. The gene sets in the library seem to be curated differential expression analysis results from the literature. Therefore, the GSA result with the custom gene set would give insight into how similar or different our results are from that of the literature.

The details of the analysis in R markdown can be found in Appendix B.2.

3. Monocyte Analysis

In order to analyze the data, alignment was performed using Kallisto. After alignment, we performed DE analysis using *EdgeR* workflow, very similar to the T-cell RNA-seq data analysis. The library sizes were normalized and genes were filtered out based on counts. In this filtering step, almost 50% of the genes were removed. The design matrix was created, with only the condition of the patient included, healthy donors vs COVID-19 patients. While it is assumed that there would be a lot of confounders present, we were unable to include them in the analysis due to the unavailability of patient information. Then the dispersion was calculated and the quasi-likelihood test was performed. Finally, the *P-values* was adjusted for multiple testing using the *Benjamini-Hochberg* method and *FDR* values were obtained.

Additionally, gene set analysis, specifically ORA using *WebGestaltR* package, with lowered *FDR* threshold was performed due to the large amount of differentially expressed genes found from the DE analysis. Two different analyses were performed, one with the KEGG pathway database and the other with a custom gene set. For the KEGG pathway database, an *FDR* threshold of 0.01 was used whereas for the custom gene set, the default threshold of 0.05 was used. The custom gene set is the same as the one used for the T-cell RNA-seq ORA from the previous section.

The details of the analysis in R markdown can be found in Appendix C.2.

4. ChIP-seq Analysis

Before beginning the analysis, MACS (Robinson et al., 2009) was used for peak calling. Here, two different

approaches were used: all the replicates of each group were pooled into a single peak for the statistical analysis; and each sample was peak called separately for the differential enrichment analysis. Therefore, the first approach gave two peaks whereas the second approach gave five.

The two peaks, obtained by pooling all replicates from each group, were then preprocessed to be converted to 'bed' file formats. Then, *GRanges* objects were created and overlaps and annotations were obtained. From there, we were able to obtain the genes of interest.

However, while the previous analysis gives genes of interest for each peak, we cannot compare the genes of interest between the mock infection and SARS-CoV-2 infection. Therefore, differential enrichment analysis using *DiffBind* was performed, and the five separately called peaks were used. Here, the block effect of the replicates was also taken into account. During the analysis, the greylist was not calculated due to an unexplainable error in the function. This error was replicated by other groups and the counselors. "Greylist" contains the signal artifacts that are cell or cell-line specific (CRUK, n.d.). As the data of this experiment comes from cell lines, analysis without the removal of greylist areas would most likely affect the downstream analysis. The result was viewed again in UCSC Genome Browser to observe peaks at genes or regions of interest.

The details of the analysis can be found in Appendix D.2.

IV. Results

1. Infinium Analysis

It was investigated whether the overall level of methylation between the groups of samples was different, averaged over all investigated sites. As per a *Welch two-sided t-test*, this was not the case (Figure 1). This indicates that there is no effect of the SARS-CoV-2 infection on the general methylation process. Additionally, this shows that our samples are fit for direct comparison at the probe level without any normalization.

Only one genic site was significantly differentially methylated between the severe and negative groups. This site was contained within the open reading frame of *FAM38A*. Its degree of methylation was higher in the patient group than in the control group. This gene codes for a mechanosensitive ion channel and has been suggested to influence how macrophages and dendritic cells react to external stimuli (Lee et al., 2022).

No other sites were significantly differentially methylated. This could possibly be due to the limited amount of samples that were analyzed, as we only used 10 samples.

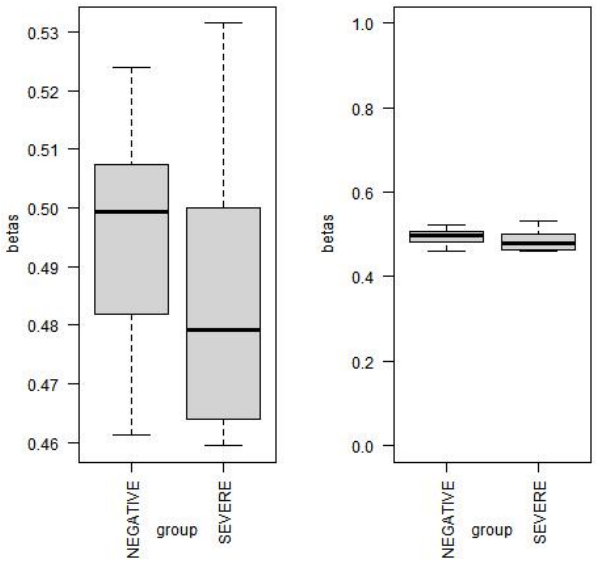


Figure 1: Overall methylation averages between COVID-19 patients with severe symptoms and negative, healthy donors.

2. CD4+ Treg Cell Analysis

In the alignment of reads, a large fraction of them, up to almost half the reads, have not been aligned (see Figure S1). This may be due to the rRNA sequences or custom adapter sequences, which were not removed prior to alignment. As mentioned above, a lot of over-represented sequences were found, and these may also contribute to poor alignment. It may also be due to some contaminants in the samples, as samples with quite a low quality were selected (purity >65%).

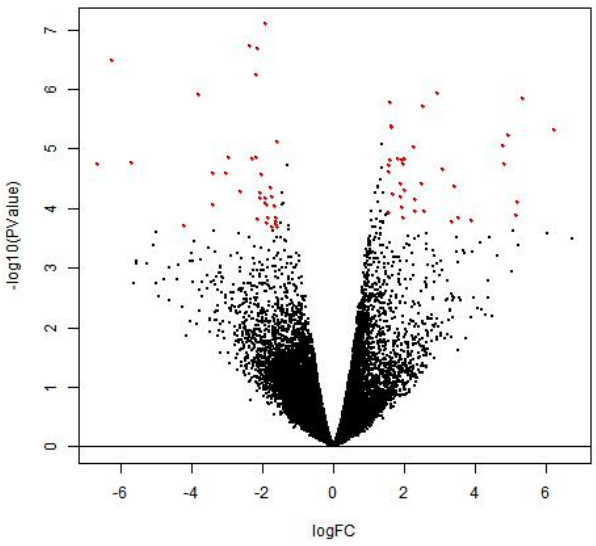


Figure 2: Volcano plot of differential expression analysis for Treg cell RNA-seq data.

The differential expression (DE) analysis of Treg cell RNA-seq data gave 95 differentially expressed genes (*FDR* <0.05). There seem to be similar amounts of over- and under-expressed genes (Figure 2), although there are slightly more overexpressed ones.

The top 10 differentially expressed genes are reported in Table 2. Some genes of note are *TCF7* and *LINC00402*. The gene *TCF7* is predominant in T-cells and is critical for natural killer cells (NCBI, n.d.) whereas *LINC00402* is a long non-coding RNA that has been recently discovered as a possible regulator for T-cells (Peltier et al., 2018). Many other genes in the top 10 are involved in the cell cycle, such as gene *RGCC*, which regulates cell cycles. Cell cycle regulation is also affected in viral infections, as viruses utilize and manipulate cell cycles to form a favourable environment for replication (Fan et al., 2018). Therefore, some of these genes may also be differentially expressed due to the SARS-CoV-2 infection.

Table 2: *Top differentially expressed genes from Treg cell RNA-seq analysis.*

Gene symbol	*logFC*	*FDR*
TCF7	-1.932328	0.001515181
RGCC	-2.372796	0.001515181
IQCN	-2.147297	0.001515181
XIST	-6.279352	0.001840419
LINC00402	-2.202009	0.002527614
UBE2C	2.913994	0.003801939
PLK2	-3.829977	0.003801939
PPBP	5.325215	0.003964030
JPT1	1.574384	0.004100106
TK1	2.507733	0.004271379

Gene set analyses were performed to gain more insight into the types of genes that were within this group and to compare our results to others. When custom gene sets were used, three gene sets out of eight were almost fully overlapping with our results (see Figure S5). Therefore, we can conclude that the results of our analysis do not fall too far from that of others in the literature.

A more insightful gene set analysis was obtained with the use of KEGG pathway database where a number of pathways involved in viral infection were found (Table 3). *Endocytosis*, *lysosome*, *autophagy* are involved in removing contaminants, such as viral molecules, and *viral carcinogenesis* is involved in viral infections, hence its name. Other pathways may also be involved in SARS-CoV-2 infections even without direct connections. For instance, *RNA transport* may be involved due to SARS-CoV-2 being an RNA virus.

The list of pathways in Table 3 consists of the ones selected based on the weighted set cover as the redundancy reduction method. The weighted set cover method subsets the gene sets so that the minimum number of gene sets that can cover as many enriched genes as possible are given (Liao et al., 2019). A more complete list can be found in Table S1 (Appendix B.3).

Table 3: *Top 10 gene sets based on weighted set cover from Treg cell RNA-seq analysis.*

Gene Set	Description
hsa03040	Spliceosome
hsa04141	Protein processing in endoplasmic reticulum
hsa04144	Endocytosis
hsa04714	Thermogenesis
hsa04142	Lysosome
hsa03013	RNA transport
hsa04140	Autophagy
hsa04120	Ubiquitin mediated proteolysis
hsa05203	Viral carcinogenesis
hsa00230	Purine metabolism

3. Monocyte Analysis

During alignment, almost half of the reads have not been aligned (see Figure S6), which may have two main explanations. It may be due to the rRNA sequences, which were not removed from the data prior to alignment. It may also be due to errors during the purification and filtering steps, where contaminants from the peripheral blood may not have been properly removed.

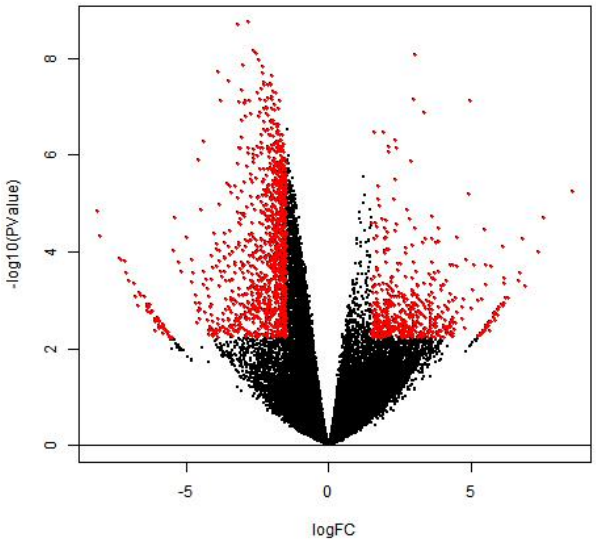


Figure 3: Volcano plot of differential expression analysis for monocyte RNA-seq data.

The differential expression analysis of monocyte

RNA-seq data gave 3,916 differentially expressed genes (*FDR* <0.05). Among these differentially expressed genes, more underexpressed genes were found in COVID-19 patients compared to human donors (Figure 3). The top 10 differentially expressed genes with gene symbols are reported in Table 4. Some genes to note are *GIMAP4*, *NEAT1*, and *GIMAP8*. *GIMAP4* and *GIMAP8* are in the family of genes for "GTPase immune-associated proteins" (*Gimpa*), and therefore are involved in immune responses (Ciucci & Bosselut, 2014). *NEAT1* is a long coding RNA that activates inflammasomes in macrophages (P. Zhang et al., 2019).

Table 4: *Top differentially expressed genes from monocyte RNA-seq analysis.*

Gene symbol	*logFC*	*FDR*
GIMAP4	-2.852291	3.286572e-05
SMG1P5	-3.192360	3.286572e-05
ZNF638	-2.672108	5.698432e-05
NEAT1	-2.569576	5.698432e-05
PIM1	3.002630	5.698432e-05
ATRX	-2.454092	5.806329e-05
ZNF518A	-3.037333	6.100015e-05
NBPF10	-2.309884	6.100015e-05
VPS13C	-2.020899	6.899340e-05
GIMAP8	-3.535728	7.060978e-05

Due to the abundance of differentially expressed genes, we performed a gene set analysis with lowered *FDR* threshold, as discussed above. From the custom gene set analysis, we find that out of the 8 sets that were used, 3 gene sets overlap almost fully with the differentially expressed genes (see Figure S10). As the gene sets that overlap with our result consists of differentially expressed genes from other studies, we can conclude that our results do not fall so far from others in the literature.

From the gene set analysis with KEGG pathway database, we find a number of pathways involved in infection and inflammation. Table 5 shows the result with the weighted set cover as the redundancy reduction method. Among these, *endocytosis* and *lysosome* are involved in the immune response during infection. *human T-cell leukaemia virus 1 infection* is involved in viral infection, and therefore, some genes in the pathway may most likely be involved in other viral infections. *MAPK signaling pathway* is involved in inflammatory responses and apoptosis (W. Zhang & Liu, 2002) and has already been linked to SARS-CoV-2 infection in the past (Goel et al., 2021). In addition, pathways involved in protein production are also overrepresented, potentially indicating high cellular activity.

The full result without redundancy reduction can be found in Table S2.

Table 5: *Top 10 gene sets based on weighted set cover from monocyte RNA-seq analysis.*

Gene Set	Description
hsa04144	Endocytosis
hsa04360	Axon guidance
hsa05166	Human T-cell leukemia virus 1 infection
hsa04714	Thermogenesis
hsa04142	Lysosome
hsa03010	Ribosome
hsa04141	Protein processing in endoplasmic reticulum
hsa04510	Focal adhesion
hsa04010	MAPK signaling pathway
hsa03040	Spliceosome

4. ChIP-seq Analysis

Initial analysis showed that the peaks of the SARS-CoV-2 infected and mock-infected groups were different, with 128 different genes between the two groups. Furthermore, by viewing the peaks in the UCSC Genome Browser, we see that there are indeed regions that have differences. However, this is a simple peak statistical analysis, without any differential analysis.

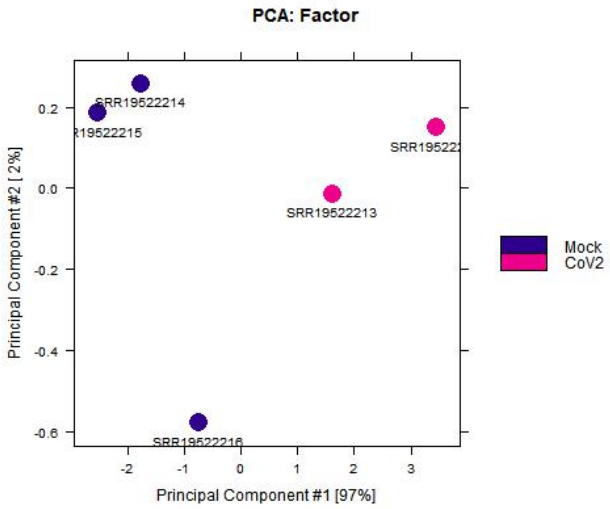


Figure 4: PCA result from ChIP-seq data analysis.

Therefore, we performed differential enrichment analysis, and saw a separation between the mock treatment and the SARS-CoV-2 treatment (Figure 4 and 5).

6 sites were found with a differential level (*FDR* <0.05) of the H3K9ac modification (Table 6). For each of these, the level of the modification was lower in the SARS-CoV-2 infected cells than in the mock-infected negative control (Figure 5). As H3K9ac modification is commonly involved in the activation of gene regulatory elements (Karmodiya et al., 2012), a lower overall level

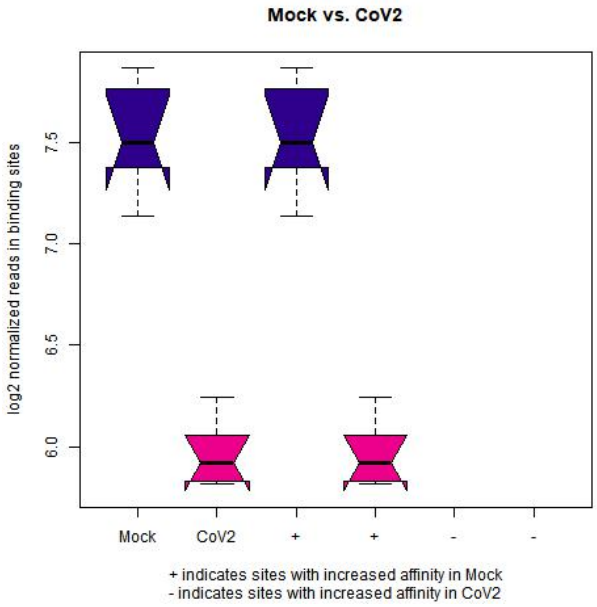


Figure 5: Box plot of histone modifications from ChIP-seq data analysis.

of this modification would suggest that, overall, genes are less active. Thus, our results imply a lower level of transcription of these 6 genes in the SARS-CoV-2 infected cells.

Table 6: *Significant genes from differential enrichment analysis of ChIP-seq data.*

Gene symbol
CEBPB
LOC105377730
DUSP1
FTL
DDIT4
TRIB3

Some genes of note are *CEBPB*, *DUSP1*, *FTL*, *DDIT4*, and *TRIB3*. *CEBPB* codes for a protein of the same name, which regulates genes involved in inflammation and other immune responses (NCBI, n.d.). *DUSP1* is involved in the *MAPK signaling pathway* and in immune responses whereas *FTL* can stimulate the interleukin-4 production (Y. Zhang et al., 2022). *DDIT4* is involved in response to viral infections and *TRIB3* can sensitize cells for apoptosis (NCBI, n.d.).

Furthermore, we viewed the peaks without the *FDR* threshold in the UCSC Genome Browser after the differential enrichment analysis. We focused on genes and regions found in the two previous RNA-seq analysis results. Specifically, we focused on the top 10 genes from each T-cell RNA-seq analysis and monocyte RNA-seq analysis results. We found the genes *JPT1*, *PLK2*, *TCF7*, *UBE2C*, *NEAT1*, *ZNF518A*, and *PIM1* had a difference in peaks between the mock-infected and SARS-CoV-2 infected cell lines out of the 20 we viewed.

V. Discussion

The aim of this study was to integrate omics data related to the pathology of SARS-CoV-2. To this end, we analyzed methylation profiling data from whole blood samples, RNAseq data from T-cells, RNAseq data from monocytes, and ChIP-seq data from cell lines.

The methylation profiling yielded only one site that is differentially methylated between the group of COVID-19 patients and the group of healthy donors, the *FAM38A* gene. This could possibly be due to the limited amount of samples that were analyzed. The dataset contained 500 samples, whereas we used 10. Perhaps more significantly differentiated sites would have been identified, were we to repeat the analysis with a larger amount of samples.

The two RNA-seq analyses both investigated the transcriptomic profiles of immune cells in the blood. One concerned itself with a cell type involved in the innate immune response, the monocytes, whereas the other looked at a part of the adaptive immune system, the Treg cells.

For the T-cell analysis, we identified 95 differentially expressed transcripts. The relatively small nature of this number can be attributed to the low number of samples that were used for the analysis, or to the large number of filtered-out genes during the analysis due to low counts. Aside from the small number of samples, the data itself seemed to lack sequencing depth, with samples containing less than the ideal 10 million reads (Figure S1). Instead, the number of reads for each sample varied from 1 million to 7 million, showing high variation as well. This implies that many transcripts were missed and that counts might be artificially low. This suboptimal number of reads may have affected the analysis. It seems plausible that our result contains a multitude of false negatives. Therefore, while the analysis gave genes involved in immune responses and other related processes, it is rather difficult to fully trust our results and to reach sound biological conclusions.

For the monocytes, the analysis yielded 3,916 differentially expressed genes. Again, there were some issues with the sequencing depth as about half the samples had fewer than 10 million reads and there were also some issues with the variation in the number of reads per sample (Figure S6). As such, the same can be said about the dubiousness of the result for the monocytes as the T-cell RNA-seq analysis.

Both of the RNA-seq results were subjected to gene set analyses. From this, it could be observed that multiple pathways were overrepresented in both sets of data. Some of these were endocytosis, the lysosome, the spliceosome, and protein processing in the endoplasmatic reticulum. The former two are involved in the immune response, whereas the latter two are related to protein production which indicates a change in cellular

activity.

The ChIP-seq analysis yielded a list of 6 genic loci that were depleted in the H3K9ac epigenetic modification for SARS-CoV-2 treated cell lines. One of these is an undescribed gene, however, the others have functions that are involved in the immune response.

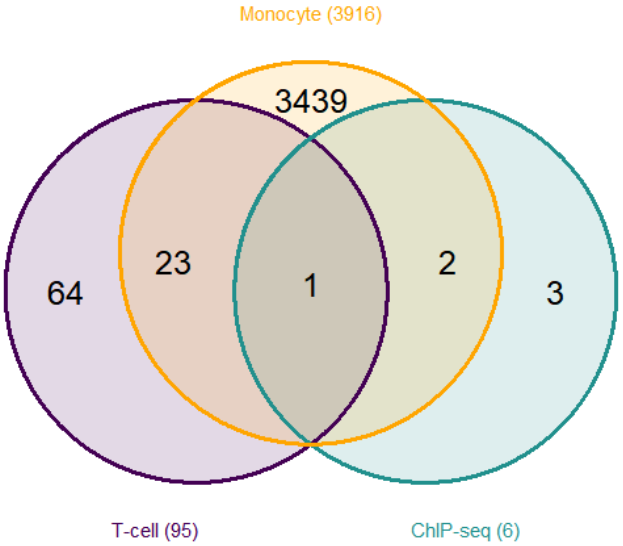


Figure 6: Venn diagram of the differentially expressed genes from the T-cell RNA-seq analysis, monocyte RNA-seq analysis, and ChIP-seq analysis.

These results were generated in different experiments and/or by the use of different platforms. This makes direct comparison difficult. However, they still correspond to one another at some level. For instance, some epigenetically modified or differentially expressed genes are shared between the ChIP-seq and RNA-seq experiments (Figure 6). The one gene that is present in all analyses is *DUSP1*. As mentioned previously, this gene is involved in the MAPK pathway, which has already been linked to the cellular response in a SARS-CoV-2 infection. *FAM38A*, the only differential genic methylation site from the Infinium analysis, was not found in any of the other analyses.

In this study, we do not find a lot of overlapping differentially expressed or enriched genes. It may be due to the fact that the data we use are of varying quality, with most yielding few significant differential genes or transcripts. Alternatively, the differences can also be due to biological variation. For instance, the same transcript might be significantly overexpressed in T-cells but not in monocytes following the viral infection. Moreover, a decrease in the levels H3K9ac epigenetic modification of a gene may not necessarily correspond to higher transcriptional activity.

As stated before, this direct comparison between the different results is suboptimal. It can be difficult to differentiate to which extent differences between the results are due to technical variation or due to the use of different methodologies. Instead of a direct comparison, multi-omics data would, in a more complex and full multi-omics analysis, be integrated using different mathematical methodologies, such as network-based integration or unsupervised clustering integration (Subramanian et al., 2020).

VI. Conclusion

In this report, data generated by various omics approaches were analyzed to gain insight into the behaviour of cells upon infection by SARS-CoV-2. Transcriptomic data generated by RNA-seq yielded a collection of genes and cellular pathways that were enriched in both monocytes and T-cells. Epigenetic data was used to attempt to infer the regulation of gene expression upon infection. From these analyses, we found that the gene *DUSP1*, already implicated in SARS-CoV-2 infection previously (Goel et al., 2021), is seen to be differentially expressed and enriched for RNA-seq and ChIP-seq analyses. However, the different data analysis results were not reliable and did not allow for a proper comparison between different omics analysis methods. This is due to a number of factors, such as low sequencing depth as well as low sample sizes. Regardless, some genes were represented in multiple analyses. Considering that different cell types were studied, one may tentatively state that these genes, such as the gene *DUSP1*, may be involved in a general response to SARS-CoV-2 infection. With a higher sample size and higher quality data, it might be possible to gain more knowledge from such a comparative omics analysis. In addition, it might be fruitful to conduct such an analysis on multi-omics data with a proper data integration method. This would allow for a more in-depth study of multi-omics data and allow for a more confident analysis result.

VII. Acknowledgements

The authors would like to thank the supervisor and counsellors for their support and guidance during this project.

VIII. Contributions

The preprocessing and data analysis of Infinium data was done by Y.P. and E.A. The preprocessing of T-cell RNA-seq data was done by Y.P. and E.A. and the data analysis was done by Y.P. The preprocessing and data analysis of monocyte RNA-seq data and ChIP-seq data

were done by Y.P. The writing of the report was done by Y.P. and E.A.

References

Andrews, S. (2010). *FastQC: A quality control tool for high throughput sequence data.* Retrieved from `https://www.bioinformatics.babraham.ac.uk/projects/fastqc/`

Barturen, G., Carnero-Montoro, E., Martínez-Buenoand, M., Rojo-Rello, S., Sobrino, B., Óscar Porras-Perales, ... Alarcón-Riquelme, M. E. (2022). Whole blood DNA methylation analysis reveals respiratory environmental traits involved in COVID-19 severity following SARS-CoV-2 infection. *Nature Communications*, *13*. doi: 10.1038/s41467-022-32357-2

Bray, N. L., Pimentel, H., Melsted, P., & Pachter, L. (2016). Near-optimal probabilistic RNA-seq quantification. *Nature Biotechnology*, *34*, 525-527. doi: 10.1038/nbt.3519

Brunetta, E., Folci, M., Bottazzi, B., Santis, M. D., Gritti, G., Protti, A., ... Mantovani, A. (2021). Macrophage expression and prognostic significance of the long pentraxin PTX3 in COVID-19. *Nature Immunology*, *22*, 19-24. doi: 10.1038/s41590-020-00832-x

Ciucci, T., & Bosselut, R. (2014). Gimap and T cells: a matter of life or death. *Eur J Immunol.*. doi: 10.1002/eji.201344375

CRUK. (n.d.). *GreyListChIP.* Cancer Research UK Cambridge Institute. Retrieved from `https://www.cruk.cam.ac.uk/core-facilities/bioinformatics-core/software/greylistchip`

Daamen, A. R., Bachali, P., Owen, K. A., Kingsmore, K. M., Hubbard, E. L., Labonte, A. C., ... Lipsky, A. C. G. . P. E. (2021). Comprehensive transcriptomic analysis of COVID-19 blood, lung, and airway. *Scientific Reports*, *11*. doi: 10.1038/s41598-021-86002-x

Ewels, P., Magnusson, M., Lundin, S., & Käller, M. (2016). MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics*, *32*(19), 3047-3048. doi: 10.1093/bioinformatics/btw354

Fan, Y., Sanyal, S., & Bruzzone, R. (2018). Breaking Bad: How Viruses Subvert the Cell Cycle. *Front. Cell. Infect. Microbiol.*, *19*. doi: 10.3389/fcimb.2018.00396

Galván-Peña, S., Leon, J., Chowdhary, K., Michelson, D. A., Vijaykumar, B., Yang, L., ... Thomas, M. F. (2021). Profound Treg perturbations correlate with COVID-19 severity. *Proceedings of the National Academy of Sciences*, *118*(37), e2111315118. doi: 10.1073/pnas.2111315118

Goel, S., Saheb Sharif-Askari, F., Saheb Sharif Askari, N., Madkhana, B., Alwaa, A. M., Mahboub, B., ... Halwani, R. (2021). SARS-CoV-2 Switches 'on' MAPK and NFB Signaling via the Reduction of Nuclear DUSP1 and DUSP5 Expression. *Frontiers in Pharmacology*, *12*. doi: 10.3389/fphar.2021.631879

Karmodiya, K., Krebs, A. R., Oulad-Abdelghani, M., Kimura, H., & Tora, L. (2012). H3K9 and H3K14 acetylation co-occur at many gene regulatory elements, while H3K14ac marks a subset of inactive inducible promoters in mouse embryonic stem cells. *BMC Genomics*, *13*. doi: 10.1186/1471-2164-13-424

Kee, J., Thudium, S., Renner, D. M., Glastad, K., Palozola, K., Zhang, Z., ... Korb, E. (2022). SARS-CoV-2 disrupts host epigenetic regulation via histone mimicry. *Nature*, *610*, 381–388. doi: 10.1038/s41586-022-05282-z

Kuleshov, M. V., Stein, D. J., Clarke, D. J., Kropiwnicki, E., Jagodnik, K. M., Bartal, A., ... Ma'ayan, A. (2020). The COVID-19 Drug and Gene Set Library. *Patterns*, *1*(6), 100090. doi: 10.1016/j.patter.2020.100090

Lee, M., Du, H., Winer, D. A., Clemente-Casares, X., & Tsai, S. (2022). Mechanosensing in macrophages and dendritic cells in steady-state and disease. *Frontiers in Cell and Developmental Biology*, *10*. doi: 10.3389/fcell.2022.1044729

Liao, Y., Wang, J., Jaehnig, E. J., Shi, Z., & Zhang, B. (2019). WebGestalt 2019: gene set

analysis toolkit with revamped UIs and APIs. *Nucleic Acids Research*, *47*(W1), W199-W205. doi: 10.1093/nar/gkz401

NCBI. (n.d.). *Gene.* Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information. Retrieved from `https://www.ncbi.nlm.nih.gov/gene`

Peltier, D., Robine, N., Hou, G., & Reddy, P. (2018). RNA-Seq of Human T Cells after BMT Identifies Linc00402 As a Novel Regulator of Human T Cell Alloimmunity. *Blood*, *132*(Supplement 1), 4517-4517. doi: 10.1182/blood-2018-99-113017

Robinson, M. D., McCarthy, D. J., & Smyth, G. K. (2009). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, *26*, 139-140. doi: 10.1093/bioinformatics/btp616

Subramanian, I., Verma, S., Kumar, S., Jere, A., & Anamika, K. (2020). Multi-omics Data Integration, Interpretation, and Its Application. *Bioinformatics and Biology Insights*, *14*. doi: 10.1177/1177932219899051

WHO. (n.d.). *WHO Coronavirus (COVID-19) Dashboard.* World Health Organization. Retrieved 2022-12-10, from `https://covid19.who.int/`

Zhang, P., Cao, L., Zhou, R., Yang, X., & Wu, M. (2019). The lncRNA Neat1 promotes activation of inflammasomes in macrophages. *Nature Communications*, *10*(1495). doi: 10.1038/s41467-019-09482-6

Zhang, W., & Liu, H. T. (2002). MAPK signal pathways in the regulation of cell proliferation in mammalian cells. *Cell Research*, *12*. doi: 10.1038/sj.cr.7290105

Zhang, Y., Hao, S., Xiao, N., Zhang, Y., Wang, H., Li, L., ... Shao, Z. (2022). Ferritin Light Chain: A Candidate Autoantigen in Immuno-Related Pancytopenia. *Frontiers in Immunology*, *13*. doi: 10.3389/fimmu.2022.851096

A Infinum Data Analysis

This section gives details on the data analysis of Infinium data and contains the analysis results.

1 Load packages

```
suppressPackageStartupMessages({
  library(GEOquery)
  library(wateRmelon)
  library(dplyr)
  library(msqrob2)
  library(ChAMPdata)
})
```

2 Intensity values

```
infdata <- readEPIC("data/methylation")
```

```
dim(betas(infdata))
```

```
## [1] 865918     10
```

```
head(betas(infdata))
```

```
##            GSM5414435_204361730068_R05C01 GSM5414470_204379160004_R03C01
## cg00000029                      0.3556838                      0.2933770
## cg00000103                      0.7848721                      0.8677840
## cg00000109                      0.7975639                      0.7786260
## cg00000155                      0.8244812                      0.8626396
## cg00000158                      0.8298662                      0.8819975
## cg00000165                      0.1025613                      0.1528339
##            GSM5414478_204390590100_R08C01 GSM5414518_204674440115_R01C01
## cg00000029                      0.2724390                      0.3699747
## cg00000103                      0.7699014                      0.8598204
## cg00000109                      0.7767956                      0.8224483
## cg00000155                      0.8468829                      0.8690062
## cg00000158                      0.8822017                      0.9091085
## cg00000165                      0.1239153                      0.1303414
##            GSM5414562_204390590100_R07C01 GSM5414644_204361730067_R02C01
## cg00000029                     0.23716963                      0.2622126
## cg00000103                     0.75223376                      0.8079599
## cg00000109                     0.73103603                      0.7951461
## cg00000155                     0.79915980                      0.8355211
## cg00000158                     0.79054143                      0.8989214
## cg00000165                     0.09286153                      0.1432361
##            GSM5414687_204361730127_R04C01 GSM5414705_204379060024_R02C01
## cg00000029                     0.21083364                     0.26020282
## cg00000103                     0.78234436                     0.83657084
## cg00000109                     0.78300455                     0.80344828
## cg00000155                     0.81052443                     0.81866953
## cg00000158                     0.84782609                     0.89369202
## cg00000165                     0.08414742                     0.08248638
```

```
##            GSM5414752_204674440038_R01C01 GSM5414837_204667550003_R03C01
## cg00000029                      0.2583416                      0.3001314
## cg00000103                      0.8137255                      0.8609935
## cg00000109                      0.8111842                      0.8896016
## cg00000155                      0.8628953                      0.9003505
## cg00000158                      0.8602212                      0.9357088
## cg00000165                      0.1460130                      0.1441128
```

```
dim(exprs(infdata))
```

```
## [1] 865918     10
```

```
head(exprs(infdata))
```

```
##            GSM5414435_204361730068_R05C01 GSM5414470_204379160004_R03C01
## cg00000029                     -0.8571734                     -1.268185
## cg00000103                      1.8672628                      2.714439
## cg00000109                      1.9781339                      1.814444
## cg00000155                      2.2318608                      2.650792
## cg00000158                      2.2862087                      2.901957
## cg00000165                     -3.1293266                     -2.470681
##            GSM5414478_204390590100_R08C01 GSM5414518_204674440115_R01C01
## cg00000029                     -1.417135                     -0.7679832
## cg00000103                      1.742422                      2.6167587
## cg00000109                      1.799169                      2.2116859
## cg00000155                      2.467527                      2.7298674
## cg00000158                      2.904790                      3.3222350
## cg00000165                     -2.821716                     -2.7381535
##            GSM5414562_204390590100_R07C01 GSM5414644_204361730067_R02C01
## cg00000029                     -1.685443                     -1.492468
## cg00000103                      1.602201                      2.072876
## cg00000109                      1.442530                      1.956625
## cg00000155                      1.992436                      2.344774
## cg00000158                      1.916176                      3.152718
## cg00000165                     -3.288170                     -2.580503
##            GSM5414687_204361730127_R04C01 GSM5414705_204379060024_R02C01
## cg00000029                     -1.904224                     -1.507493
## cg00000103                      1.845756                      2.355822
## cg00000109                      1.851356                      2.031296
## cg00000155                      2.096844                      2.174660
## cg00000158                      2.478047                      3.071528
## cg00000165                     -3.444124                     -3.475502
##            GSM5414752_204674440038_R01C01 GSM5414837_204667550003_R03C01
## cg00000029                     -1.521475                     -1.221490
## cg00000103                      2.127112                      2.630849
## cg00000109                      2.103050                      3.010440
## cg00000155                      2.653907                      3.175553
## cg00000158                      2.621562                      3.863365
## cg00000165                     -2.548117                     -2.570222
```

3 Annotation

We will be investigating differential DNA methylation between samples from patients with different disease states.

```
# Read table and clean up
meth_annotation <- getGEO(filename="GSE179325_series_matrix.txt.gz")
meth_annotation <- pData(meth_annotation)
meth_annotation <- meth_annotation %>%
  dplyr::rename(Disease_state = `disease state:ch1`,
                Tissue_type = `tissue/cell type:ch1`,
                Organism = `organism_ch1`,
                Age = `age:ch1`,
                Gender = `gender:ch1`)
meth_files <- list.files("data/methylation/")
meth_files <- meth_files[grep("Grn.idat",meth_files)]
meth_samples <- unlist(strsplit(meth_files,"_") %>% vapply('[','',1))
meth_annotation <- dplyr::filter(meth_annotation, geo_accession %in% meth_samples)
```

```
# Annotation data
print(meth_annotation[c("geo_accession", "Disease_state", "Age", "Gender",
                        "Tissue_type", "Organism")])
```

```
##            geo_accession Disease_state Age Gender Tissue_type     Organism
## GSM5414435    GSM5414435      NEGATIVE  24      M Whole Blood Homo sapiens
## GSM5414470    GSM5414470        SEVERE  89      M Whole Blood Homo sapiens
## GSM5414478    GSM5414478      NEGATIVE  70      F Whole Blood Homo sapiens
## GSM5414518    GSM5414518      NEGATIVE  51      M Whole Blood Homo sapiens
## GSM5414562    GSM5414562      NEGATIVE  73      F Whole Blood Homo sapiens
## GSM5414644    GSM5414644        SEVERE  93      F Whole Blood Homo sapiens
## GSM5414687    GSM5414687        SEVERE  67      F Whole Blood Homo sapiens
## GSM5414705    GSM5414705        SEVERE  74      F Whole Blood Homo sapiens
## GSM5414752    GSM5414752      NEGATIVE  88      F Whole Blood Homo sapiens
## GSM5414837    GSM5414837        SEVERE  51      M Whole Blood Homo sapiens
```

4 Preprocessing

4.1 Remove NA values

```
infdata <- infdata[rowSums(is.na(exprs(infdata)))==0,]
```

4.2 Filtering

```
#Remove probes for which calling p-value is insufficient
infdata.pf <- pfilter(infdata)
```

```
## 0 samples having 1 % of sites with a detection p-value greater than 0.05 were removed
```

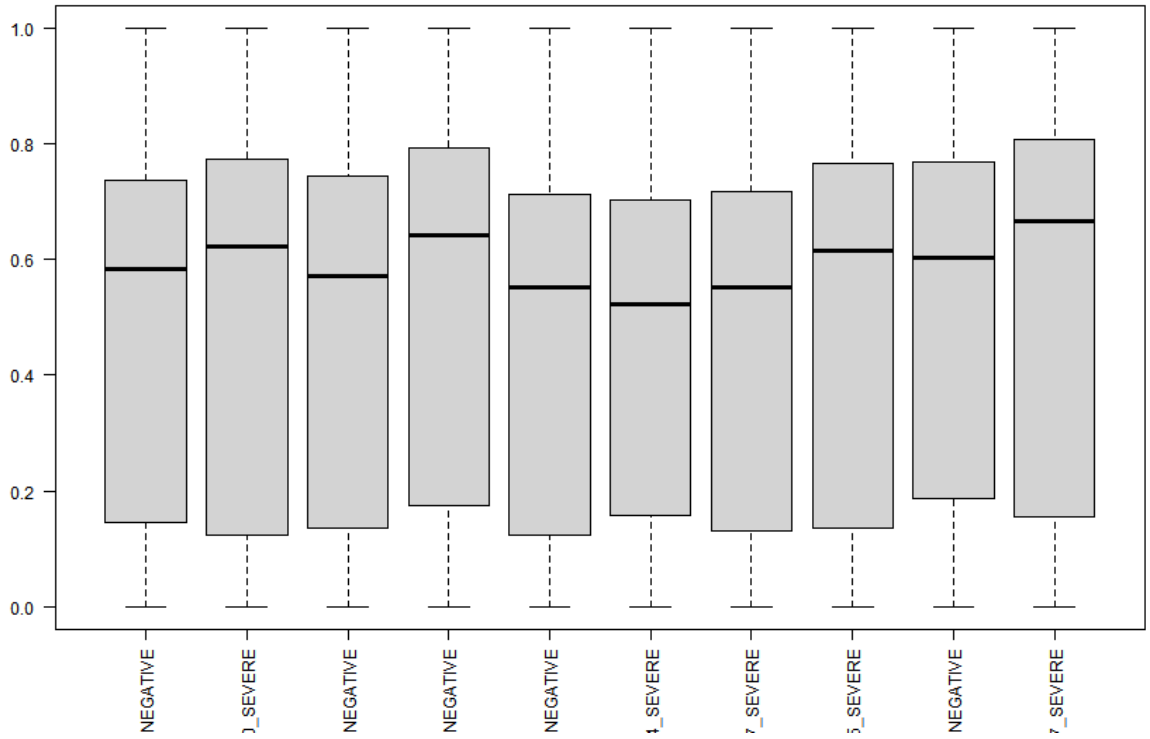
```
## Samples removed:
## 17430 sites were removed as beadcount <3 in 5 % of samples
## 0 sites having 1 % of samples with a detection p-value greater than 0.05 were removed
```

```
#change sample names
sampleNames(infdata) <- paste(meth_annotation["geo_accession"][,1],
                              meth_annotation["Disease_state"][,1],sep="_")
```

17430 sites were removed as beadcount <3 in 5% of samples. And no sites had 1% of samples with a detection p-value greater than 0.05.

4.3 Degree of methylation between groups

```
boxplot(betas(infdata),las=2)
```



There does not seem to be a big difference between groups.

We perform a t-test to confirm.

```
meth_mean_SEVERE <- rep(0,5)
meth_mean_NEGATIVE <- rep(0,5)
n <- 1
m <- 1
```

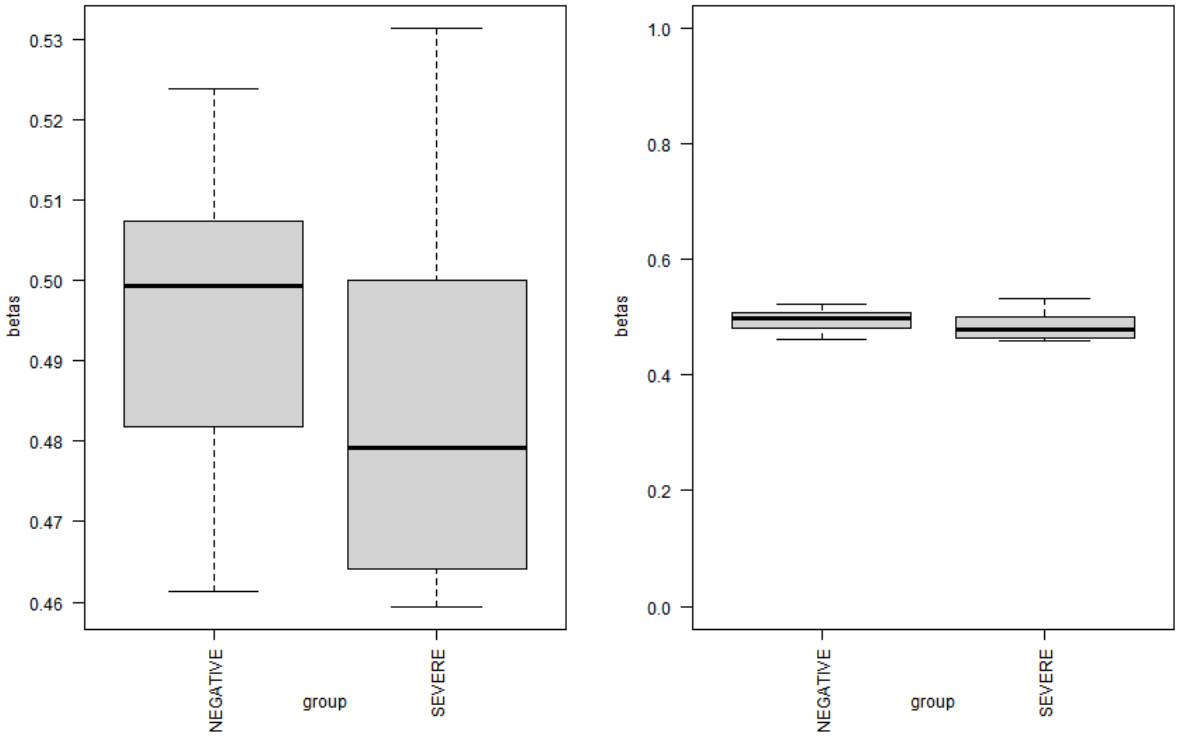
```
for (i in 1:ncol(infdata)){
  if (meth_annotation["Disease_state"][,1][i] == "SEVERE"){
  meth_mean_SEVERE[n] <- mean(betas(infdata)[,i])
  n <- n + 1
  }
  else {
  meth_mean_NEGATIVE[m] <- mean(betas(infdata)[,i])
  m <- m + 1
}
}
```

```
t_test_res <- t.test(meth_mean_SEVERE,meth_mean_NEGATIVE)
t_test_res
```

```
##
##  Welch Two Sample t-test
##
## data:  meth_mean_SEVERE and meth_mean_NEGATIVE
## t = 0.0098532, df = 7.7592, p-value = 0.9924
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.03986180  0.04020205
## sample estimates:
## mean of x mean of y
## 0.4908463 0.4906761
```

The t-test also shows the same result as the previous box plot. The p-value is 0.9924, above 0.05. Therefore, we cannot reject null hypothesis, which states that there is no difference between groups in terms of average methylation.

```
dat_boxplot <- data.frame(betas = c(meth_mean_NEGATIVE, meth_mean_SEVERE),
                                    group = meth_annotation$Disease_state)
par(mfrow=c(1,2))
boxplot(betas~group,dat_boxplot,las=2)
boxplot(betas~group,dat_boxplot,las=2,ylim=c(0,1))
```

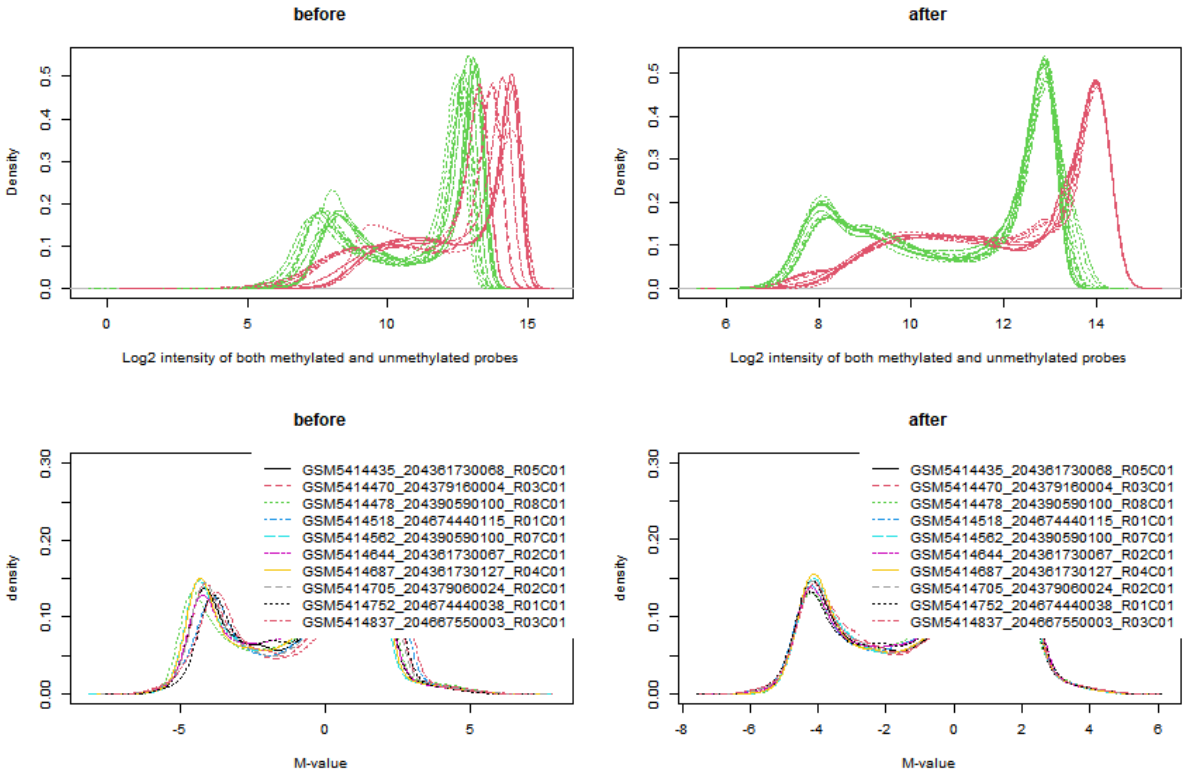


The box plots also show that there is little difference between average methylation levels between groups.

4.4 Normalisation & Quality Control

```
# Perform normalization including dye color adjustment
infdata.dasen.pf <- dasen(infdata.pf)
# Make methylumi objects to check density and color bias adjustment
infdataM <- as(infdata.pf, "MethyLumiM")
infdataN <- as(infdata.dasen.pf, "MethyLumiM")
```

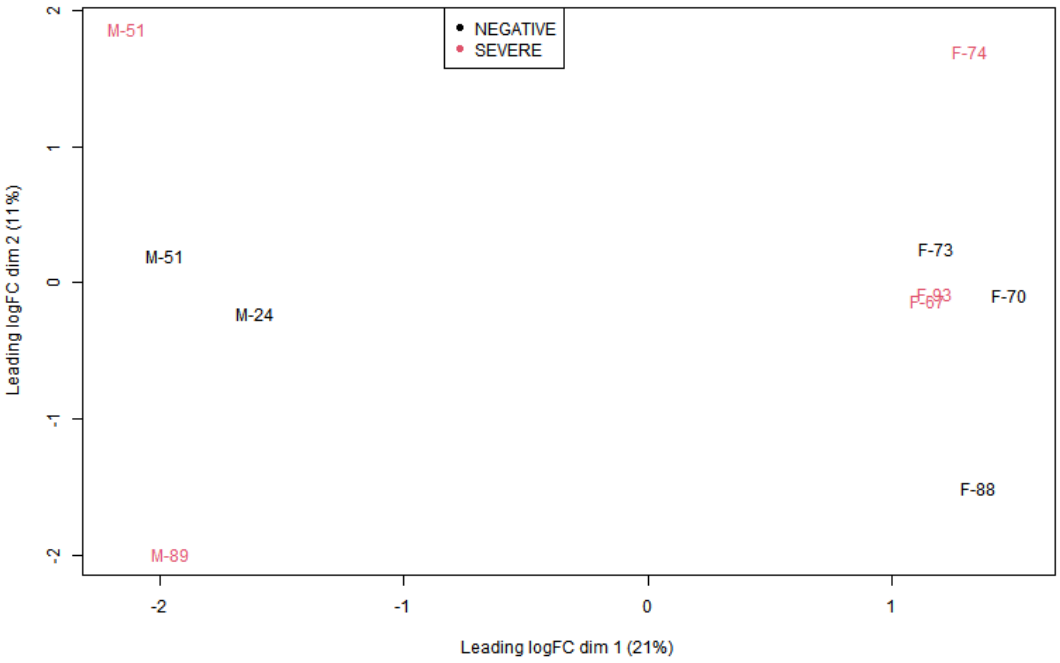
```
# Make QC plot
par(mfrow=c(2,2))
plotColorBias1D(infdataM,channel="both",main="before")
plotColorBias1D(infdataN,channel="both",main="after")
density(infdataM,xlab="M-value",main="before")
density(infdataN,xlab="M-value",main="after")
```



5 Differential analysis

5.1 MDS plot

```
disease <- as.factor(meth_annotation$Disease_state)
par(mar=c(6,6,6,6))
plotMDS(infdataN, labels=paste(meth_annotation$Gender,meth_annotation$Age,sep='-'),
        col=as.double(disease))
par(xpd=T)
legend(par("usr")[2]*-0.5,par("usr")[4]*1,sort(unique(disease)),
       pch=c(16),col=as.double(sort(unique(disease))))
```



There seems to be quite a big gender effect in the samples.

Surprisingly, there does not seem to be a big age effect. However, only a few samples are used in the analysis. And in the paper of the dataset, the authors report that they find a large age effect, larger than that of the gender effect.

Therefore, we add both gender and age as the confounding factor.

5.2 Design matrix

```
disease <- as.factor(meth_annotation$Disease_state)
age <- as.numeric(meth_annotation$Age)
gender <- as.factor(meth_annotation$Gender)
```

```
design <- model.matrix(~disease+gender+age)
design
```

```
##    (Intercept) diseaseSEVERE genderM age
## 1            1             0       1  24
## 2            1             1       1  89
## 3            1             0       0  70
## 4            1             0       1  51
## 5            1             0       0  73
## 6            1             1       0  93
## 7            1             1       0  67
```

```
## 8            1             1       0  74
## 9            1             0       0  88
## 10           1             1       1  51
## attr(,"assign")
## [1] 0 1 2 3
## attr(,"contrasts")
## attr(,"contrasts")$disease
## [1] "contr.treatment"
##
## attr(,"contrasts")$gender
## [1] "contr.treatment"
```

We do not consider interaction terms so the analysis does not become too complex.

```
cont.matrix <- makeContrast("diseaseSEVERE=0", parameterNames = colnames(design))
cont.matrix
```

```
##               diseaseSEVERE
## (Intercept)               0
## diseaseSEVERE             1
## genderM                   0
## age                       0
```

5.3 Limma

```
fit <- lmFit(infdataN,design)
fit2 <- contrasts.fit(fit,cont.matrix)
fit2 <- eBayes(fit2)
```

5.4 Results

```
# DE results
LIMMAout <- topTable(fit2,adjust="BH",number=nrow(exprs(infdataN)))
head(LIMMAout)
```

```
##              Probe_ID DESIGN COLOR_CHANNEL     logFC     AveExpr          t
## cg26748794 cg26748794      I           Red  3.305872 -3.24055911  23.713405
## cg16052052 cg16052052     II          Both -1.936868  0.09108089 -10.822310
## cg07997860 cg07997860     II          Both -2.073658 -1.10282150  -9.778239
## cg15770106 cg15770106     II          Both -1.378320 -2.46741195  -9.762194
## cg21463262 cg21463262     II          Both -3.701843 -2.28990335  -9.480922
## cg24407607 cg24407607     II          Both  4.336404 -0.64436611   9.438034
##                 P.Value    adj.P.Val         B
## cg26748794 3.371809e-10 0.0002852162 1.2390196
## cg16052052 6.957814e-07 0.2885357353 0.5238440
## cg07997860 1.787921e-06 0.2885357353 0.3510274
## cg15770106 1.815210e-06 0.2885357353 0.3480344
## cg21463262 2.375331e-06 0.2885357353 0.2936872
## cg24407607 2.476178e-06 0.2885357353 0.2850776
```

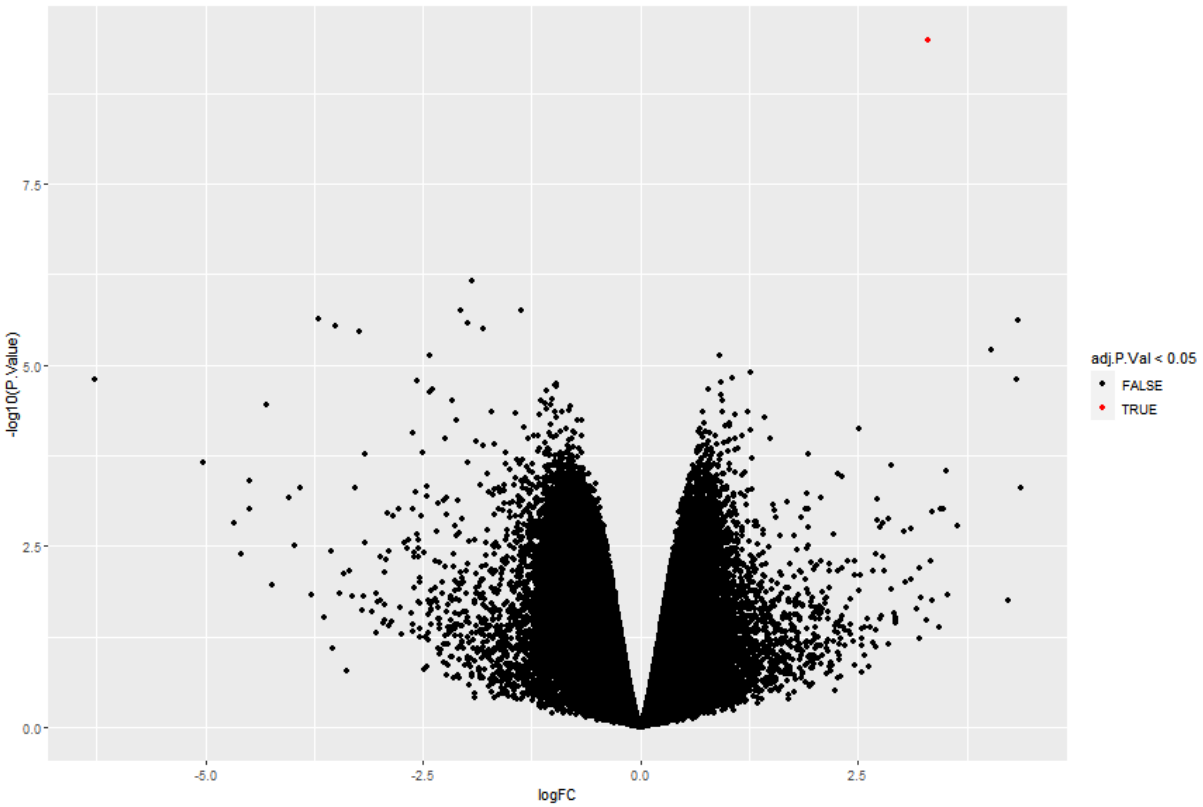
```
## Check M-values for top results
exprs(infdataN)[rownames(infdataN)%in%rownames(head(LIMMAout)),]
```

```
##            GSM5414435_204361730068_R05C01 GSM5414470_204379160004_R03C01
## cg07997860                      -2.479516                      0.8483345
## cg15770106                      -2.400828                     -2.9622571
## cg16052052                       1.308377                     -1.1934117
## cg21463262                      -4.595858                     -1.1147411
## cg24407607                      -3.677617                      2.0385855
## cg26748794                      -4.522558                     -1.7726831
##            GSM5414478_204390590100_R08C01 GSM5414518_204674440115_R01C01
## cg07997860                      -0.337780                     -0.5420765
## cg15770106                      -2.049790                     -1.7529436
## cg16052052                       1.231477                      1.2630509
## cg21463262                      -1.225879                     -1.1247784
## cg24407607                      -1.105564                     -3.1396611
## cg26748794                      -4.644392                     -4.7003296
##            GSM5414562_204390590100_R07C01 GSM5414644_204361730067_R02C01
## cg07997860                     -0.8232104                     -0.5877004
## cg15770106                     -1.7456533                     -2.8879765
## cg16052052                      0.6019459                     -0.5874813
## cg21463262                     -1.2084550                     -1.1559961
## cg24407607                     -3.3782714                      1.1558492
## cg26748794                     -4.8991325                     -1.7571803
##            GSM5414687_204361730127_R04C01 GSM5414705_204379060024_R02C01
## cg07997860                     -2.7072116                     -2.9184712
## cg15770106                     -3.2154088                     -3.2122505
## cg16052052                     -0.9132141                     -0.8719588
## cg21463262                     -4.5798357                     -5.0644461
## cg24407607                      1.4135134                      1.7610653
## cg26748794                     -1.3964143                     -1.7397103
##            GSM5414752_204674440038_R01C01 GSM5414837_204667550003_R03C01
## cg07997860                      0.9689133                     -2.4494964
## cg15770106                     -1.3954032                     -3.0516082
## cg16052052                      1.0592366                     -0.9872132
## cg21463262                      1.8958918                     -4.7249366
## cg24407607                     -3.0909734                      1.5794123
## cg26748794                     -5.3399023                     -1.6332890
```

6 Plots

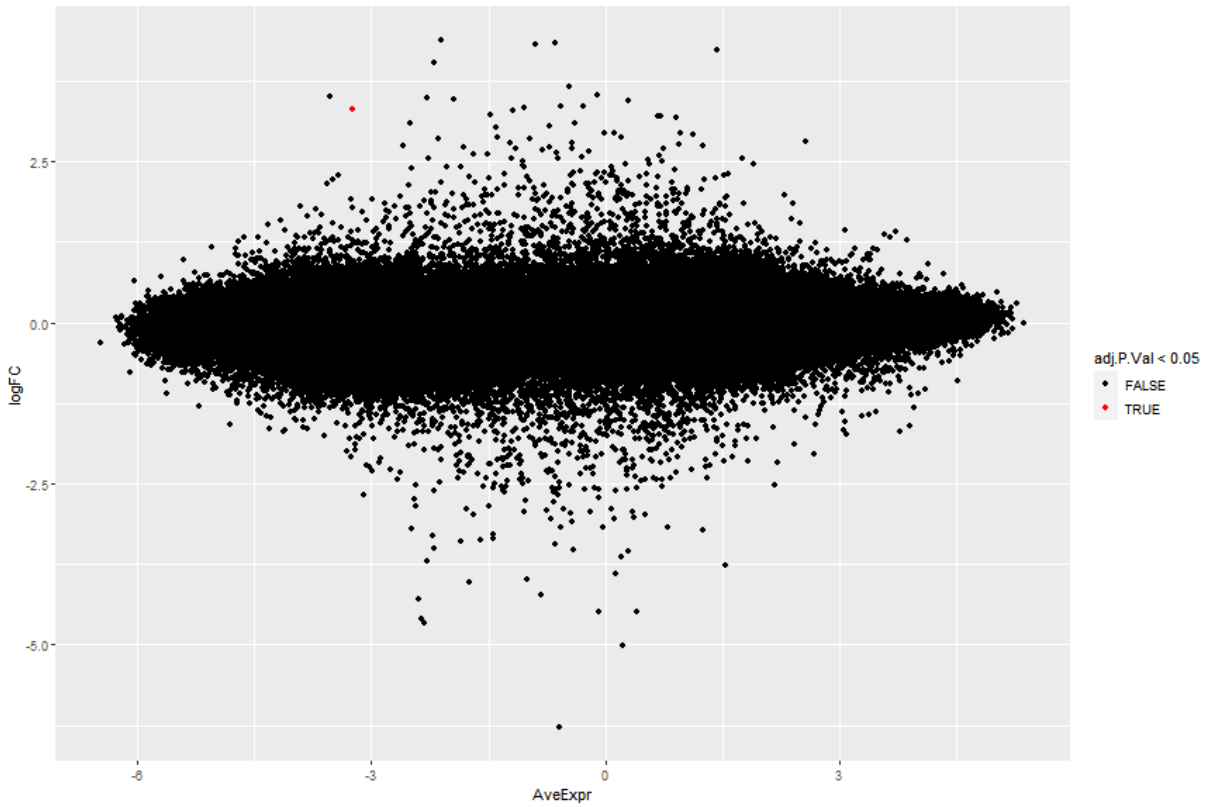
6.1 Volcano plot

```
volcano<- ggplot(LIMMAout,aes(x=logFC,y=-log10(P.Value),color=adj.P.Val < 0.05)) +
          geom_point() + scale_color_manual(values=c("black","red"))
print(volcano)
```



6.2 MA plot

```
MA<- ggplot(LIMMAout,aes(x=AveExpr,y=logFC,color=adj.P.Val < 0.05)) +
            geom_point() + scale_color_manual(values=c("black","red"))
print(MA)
```



7 Functional annotation of results

7.1 Load annotation

```
data("probe.features.epic")
annotation_MA <- probe.features
print(head(annotation_MA))
```

```
##            CHR   MAPINFO Strand Type   gene feature     cgi      feat.cgi
## cg00000029  16  53468112      F   II   RBL2 TSS1500   shore TSS1500-shore
## cg00000103   4  73470186      F   II            IGR opensea   IGR-opensea
## cg00000109   3 171916037      F   II FNDC3B    Body opensea  Body-opensea
## cg00000155   7   2590565      F   II  BRAT1    Body opensea  Body-opensea
## cg00000158   9  95010555      F   II   IARS    Body opensea  Body-opensea
## cg00000165   1  91194674      R   II            IGR   shore     IGR-shore
##                  UCSC_Islands_Name
## cg00000029 chr16:53468284-53469209
## cg00000103
## cg00000109
## cg00000155
## cg00000158
## cg00000165  chr1:91190489-91192804
##                                        SNP_ID SNP_DISTANCE
```

```
## cg00000029
## cg00000103             rs16848262;rs146966574;rs61513989      1;23;27
## cg00000109 rs572345338;rs9864492;rs553441475;rs183505630   2;17;23;24
## cg00000155                                  rs116384860           0
## cg00000158                        rs78176522;rs80089624         0;38
## cg00000165                      rs575936564;rs546501292         35;6
```

7.2 Check annotation

```
# sort probes
annotation_MA <- annotation_MA[sort(rownames(annotation_MA),index.return=T)$ix,]
# Check if all probes are present in both sets
dim(LIMMAout)
```

```
## [1] 845885      9
```

```
dim(annotation_MA)
```

```
## [1] 866895     11
```

```
sum(LIMMAout$Probe_ID%in%rownames(annotation_MA))
```

```
## [1] 845885
```

```
sum(rownames(annotation_MA)%in%LIMMAout$Probe_ID)
```

```
## [1] 845885
```

The dimensions for the annotation is not the same as our output, so we remove probes not found in our results.

```
annotation_MA <- annotation_MA[rownames(annotation_MA)%in%LIMMAout$Probe_ID,]
dim(annotation_MA)
```

```
## [1] 845885     11
```

7.3 Annotate results

```
# Sort LIMMA output alphabetically on probe name
LIMMAout_sorted <- LIMMAout[sort(LIMMAout$Probe_ID,index.return=T)$ix,]
# Add gene names to LIMMA output
LIMMAout_sorted$Gene <- annotation_MA$gene
LIMMAout_sorted$Feature <- annotation_MA$feature
LIMMAout_sorted$Chrom <- annotation_MA$CHR
LIMMAout_sorted$Pos <- annotation_MA$MAPINFO
LIMMAout_sorted$Chrom <- as.character(LIMMAout_sorted$Chrom)
LIMMAout_sorted$Gene <- as.character(LIMMAout_sorted$Gene)
LIMMAout_sorted$Feature <- as.character(LIMMAout_sorted$Feature)
```

8 Quantification of absolute methylation differences

```
dim(LIMMAout_sorted)
```

```
## [1] 845885     13
```

```
dim(betas(infdata.dasen.pf))
```

```
## [1] 845885     10
```

The dimensions are the same, so we move on.

8.1 Add gene names

```
LIMMAout_sorted$SEVERE_meth <- rowMeans(betas(infdata.dasen.pf)[rownames(infdata.dasen.pf)
    %in%LIMMAout_sorted$Probe_ID, meth_annotation["Disease_state"][,1]=="SEVERE"])
LIMMAout_sorted$NEGATIVE_meth <- rowMeans(betas(infdata.dasen.pf)[rownames(infdata.dasen.pf)
    %in%LIMMAout_sorted$Probe_ID, meth_annotation["Disease_state"][,1]=="NEGATIVE"])
LIMMAout_sorted$Abs_diff_meth <- abs(rowMeans(betas(infdata.dasen.pf)[rownames(infdata.dasen.pf)
    %in%LIMMAout_sorted$Probe_ID, meth_annotation["Disease_state"][,1]=="SEVERE"]) -
    rowMeans(betas(infdata.dasen.pf)[rownames(infdata.dasen.pf)
    %in%LIMMAout_sorted$Probe_ID, meth_annotation["Disease_state"][,1]=="NEGATIVE"]))
```

We also sort the results again on p-values.

```
LIMMAout_annot <- LIMMAout_sorted[sort(LIMMAout_sorted$P.Value,index.return=T)$ix,
                                  c(1,12,13,10,11,4,7,8,5,14,15,16)]
```

9 Interpretation

9.1 Genic regions

```
sum(LIMMAout_annot$adj.P.Val<0.05)
```

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

```
sum(LIMMAout_annot$adj.P.Val[LIMMAout_annot$Gene!=""]<0.05)
```

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

```
head(LIMMAout_annot[c(4,5,6,8,10,11,12)])
```

```
##              Gene Feature     logFC    adj.P.Val SEVERE_meth NEGATIVE_meth
## cg26748794 FAM38A    Body  3.305872 0.0002852162   0.2383889    0.03082246
## cg16052052            IGR -1.936868 0.2885357353   0.3397938    0.67462125
## cg07997860   STAC    Body -2.073658 0.2885357353   0.2816986    0.39825848
## cg15770106            IGR -1.378320 0.2885357353   0.1031013    0.21440149
## cg21463262 ATP11A   3'UTR -3.701843 0.2885357353   0.1416971    0.34529141
## cg24407607    DSE    Body  4.336404 0.2885357353   0.7400617    0.11528643
##            Abs_diff_meth
## cg26748794     0.2075664
## cg16052052     0.3348275
## cg07997860     0.1165599
## cg15770106     0.1113002
## cg21463262     0.2035943
## cg24407607     0.6247752
```

```
LIMMAout_annot_gene <- LIMMAout_annot[LIMMAout_annot$Gene!="",]
checkMeth <- LIMMAout_annot_gene %>% filter(adj.P.Val < 0.05 &
                                              abs(LIMMAout_annot_gene$logFC)>2)
checkMeth[c(4,5,6,8,10,11,12)]
```

```
##              Gene Feature    logFC    adj.P.Val SEVERE_meth NEGATIVE_meth
## cg26748794 FAM38A    Body 3.305872 0.0002852162   0.2383889    0.03082246
##            Abs_diff_meth
## cg26748794     0.2075664
```

We only find 1 differentially methylated site, at gene `FAM38A`.

9.2 Promoter regions

```
LIMMAout_annot_prom <- LIMMAout_annot_gene[grepl("TSS",LIMMAout_annot_gene$Feature) |
                                        (LIMMAout_annot_gene$Feature=="1stExon"),]
head(LIMMAout_annot_prom)
```

```
##              Probe_ID Chrom       Pos  Gene Feature      logFC      P.Value
## cg24859648 cg24859648    20  17680544 BANF2  TSS200  4.0295701 6.260505e-06
## cg05522700 cg05522700     7 120589681  ING3 TSS1500 -0.9746948 1.906641e-05
## cg15830792 cg15830792    10  64893214 NRBF2 1stExon  0.9306734 2.609545e-05
## cg02105093 cg02105093    17  27070369 TRAF4 TSS1500 -1.0865600 4.112636e-05
## cg05207622 cg05207622    10 124912266  BUB3 TSS1500 -0.9770485 4.464448e-05
## cg16565913 cg16565913     4  40632362 RBM47 TSS1500 -1.1556036 4.896532e-05
##            adj.P.Val     AveExpr SEVERE_meth NEGATIVE_meth Abs_diff_meth
## cg24859648 0.4814243 -2.19531098  0.40741562    0.08450771    0.32290791
## cg05522700 0.6846313 -3.15530390  0.05737063    0.10366787    0.04629724
## cg15830792 0.6846313 -2.80068256  0.14818353    0.09936308    0.04882045
## cg02105093 0.6846313 -1.80585646  0.16436711    0.28305028    0.11868316
## cg05207622 0.6846313 -0.09849993  0.41111762    0.52941125    0.11829363
## cg16565913 0.6846313 -1.68351593  0.18321006    0.30000328    0.11679322
```

```
head(LIMMAout_annot_prom %>% filter(LIMMAout_annot_prom$adj.P.Val<0.1))
```

```
##  [1] Probe_ID      Chrom         Pos           Gene          Feature
##  [6] logFC         P.Value       adj.P.Val     AveExpr       SEVERE_meth
## [11] NEGATIVE_meth Abs_diff_meth
## <0 > <  row.names    0 >
```

We have no differentially methylated promoter regions.

B CD4+ Treg cell RNA-seq Data Analysis

This section gives details on the data analysis of T-cell RNA-seq data and contains the analysis results.

B.1 Quality Control

FastQC was used for quality control and the FastQC results were combined via MultiQC (Ewels et al., 2016). The following shows some of the relevant figures from quality control.

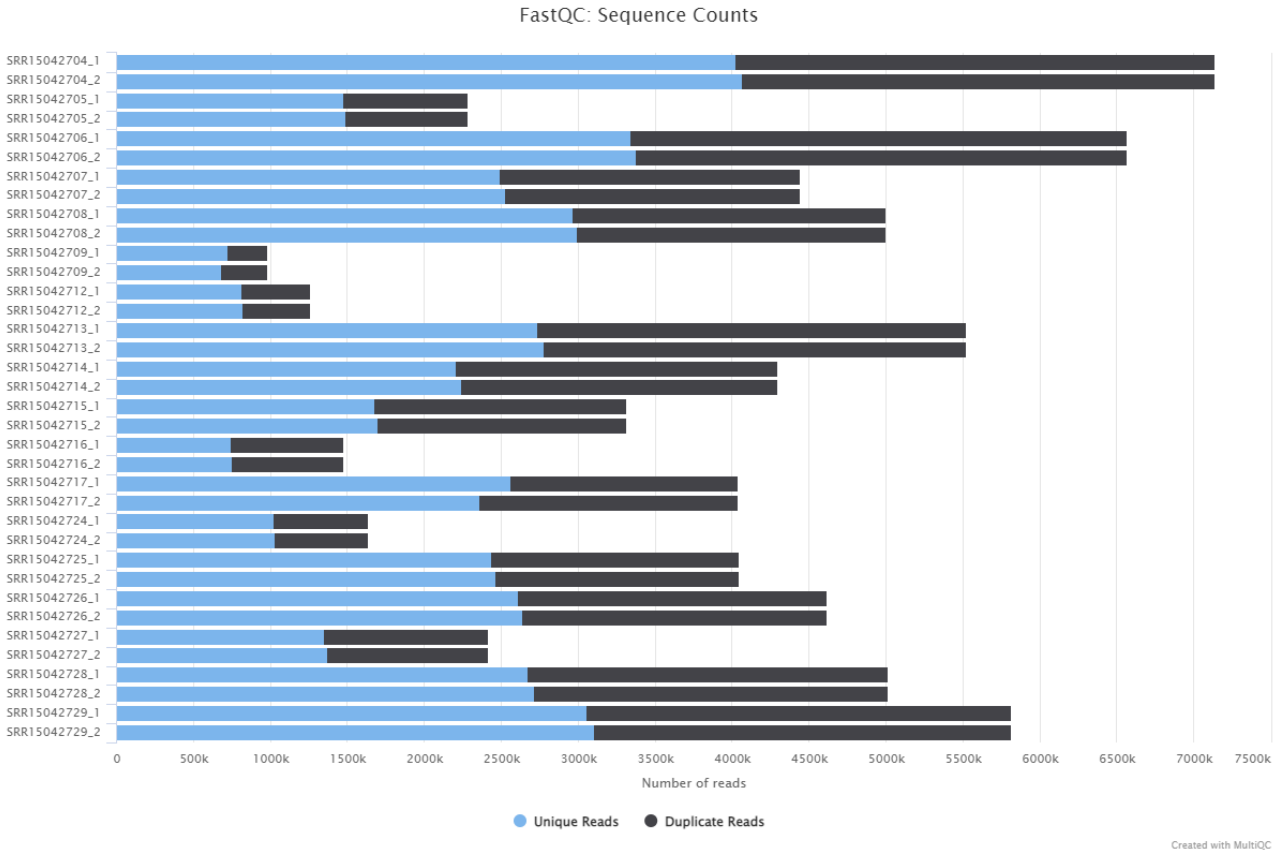


Figure S1: Sequence counts for each sample of T-cell RNA-seq data. Duplicate read counts are an estimate only. The extensions after the sample names indicate the forward read (*1*) and reverse read (*2*).

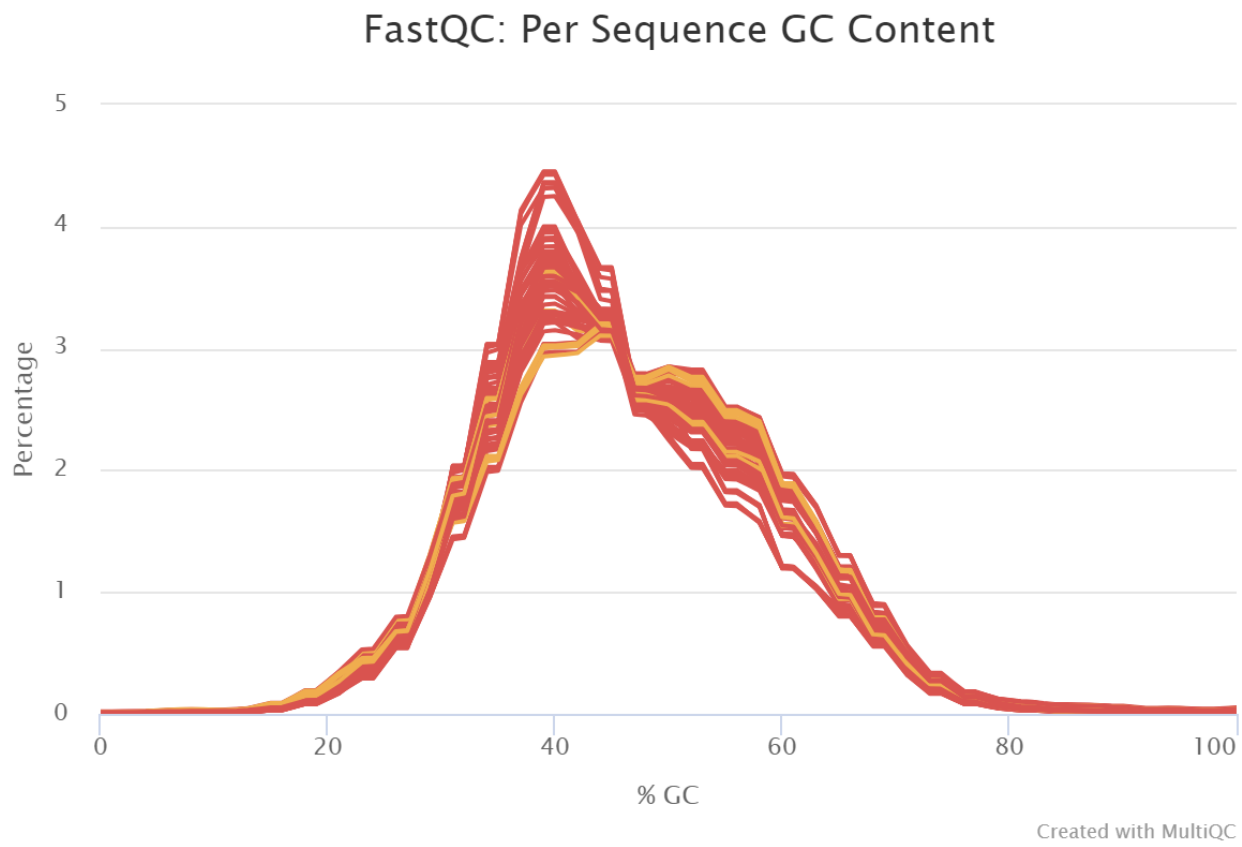


Figure S2: Per Sequence GC Content of T-cell RNA-seq data. The average GC content of reads.

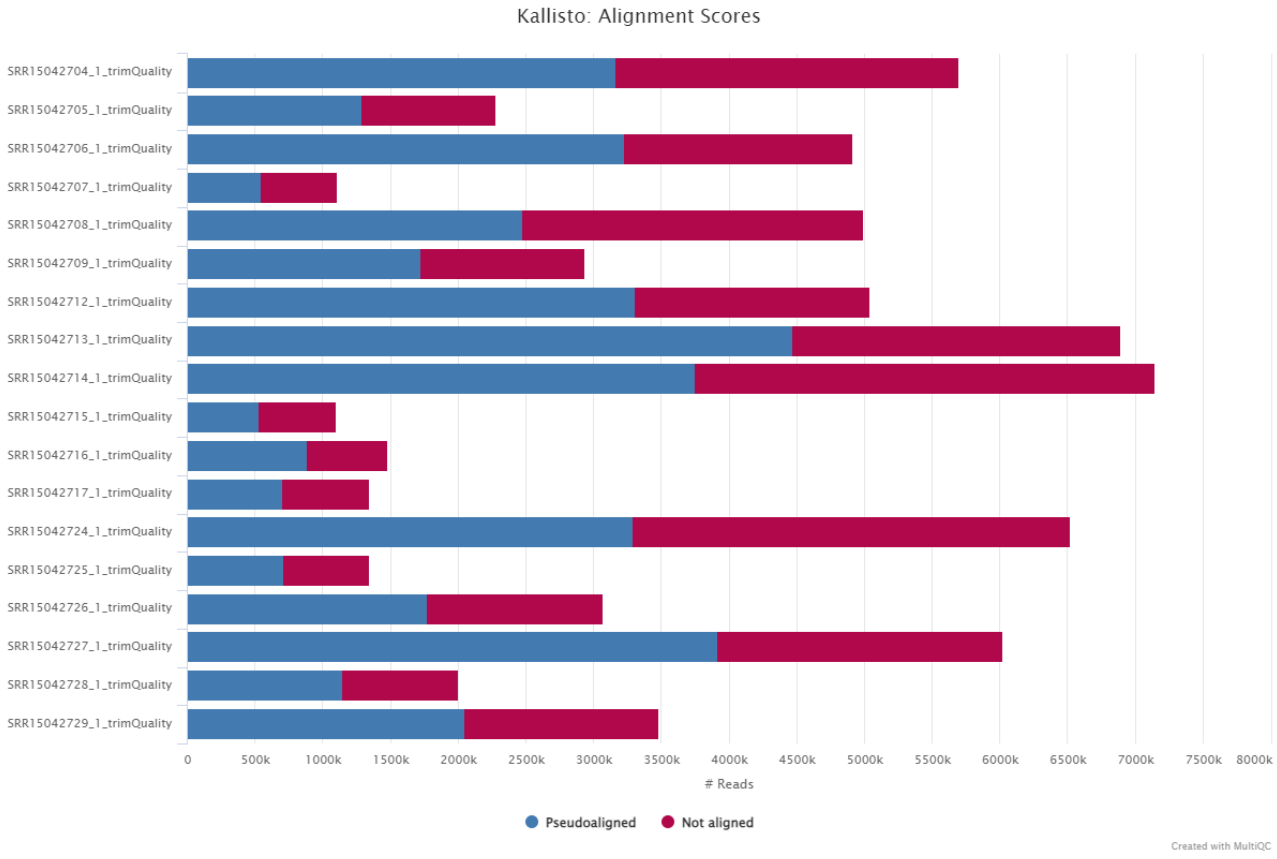


Figure S3: Kallisto alignment scores of T-cell RNA-seq data.

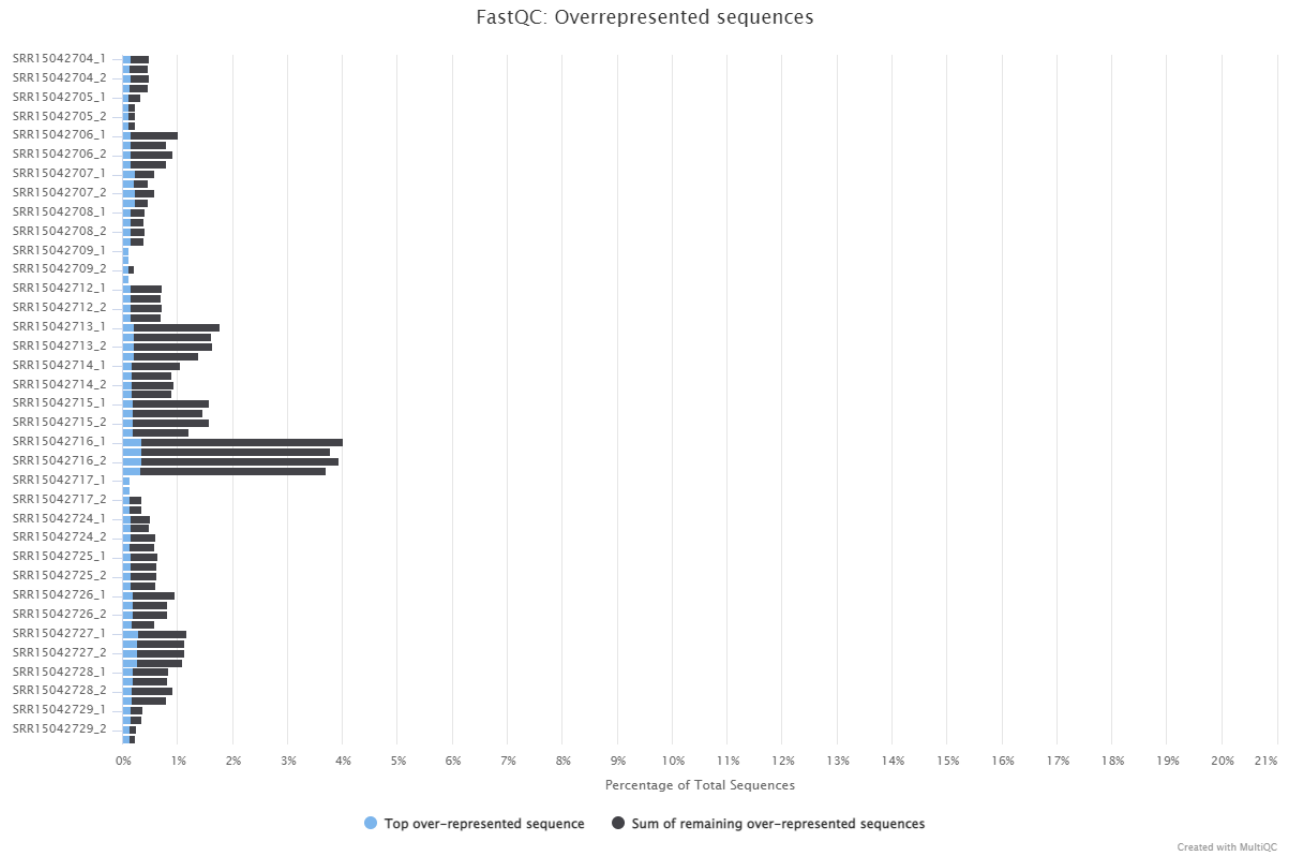


Figure S4: Overrepresented sequences of T-cell RNA-seq data. The total amount of overrepresented sequences found in each library. The extensions after the sample names indicate the forward read (*1*) and reverse read (*2*). Each sample (either forward or reverse read) shows two bars, one for the raw sample and the other one for the sample trimmed for quality.

B.2 R Markdown of Data Analysis

The following is the R markdown file of the statistical analysis of T-cell RNA-seq data.

1 Load libraries

```
suppressPackageStartupMessages({
  library(biomaRt)
  library(dplyr)
  library(tximport)
  library(edgeR)
  library(msqrob2)
  library(ggplot2)
  library(gridExtra)
  library(WebGestaltR)
})
```

2 Pseudoount table

2.1 Human annotation data

```
#annotation data
hs_annot <- useEnsembl(biomart = "ensembl", dataset = "hsapiens_gene_ensembl")

#attributes of annotation data
attributes <- listAttributes(hs_annot)

data <- getBM(attributes = c('ensembl_gene_id', 'ensembl_transcript_id',
                             'external_gene_name'), mart = hs_annot)

tx2geneGtf <- dplyr::select(data, ensembl_transcript_id, ensembl_gene_id)
tx2geneGtf <- dplyr::rename(tx2geneGtf, TXNAME = ensembl_transcript_id)
tx2geneGtf <- dplyr::rename(tx2geneGtf, GENEID = ensembl_gene_id)

head(tx2geneGtf)
```

```
##            TXNAME          GENEID
## 1 ENST00000387314 ENSG00000210049
## 2 ENST00000389680 ENSG00000211459
## 3 ENST00000387342 ENSG00000210077
## 4 ENST00000387347 ENSG00000210082
## 5 ENST00000386347 ENSG00000209082
## 6 ENST00000361390 ENSG00000198888
```

2.2 Generate pseudocount table

```
# Get file locations
tcell_files <- list.files("data/RNAseq_tcell/")
tcell_files <- tcell_files[grep("abundance.tsv",tcell_files)]
tcell_samples <- unlist(strsplit(tcell_files,"_"))[c(1:length(tcell_files))*2-1]
tcell_files <- paste(rep("data/RNAseq_tcell/",length(tcell_files)),tcell_files,sep="")
```

```
names(tcell_files) <- tcell_samples

# Load RNAseq data
tcell_txi <- tximport(tcell_files, type = "kallisto", tx2gene = tx2geneGtf)
head(tcell_txi$counts)
```

```
##                 SRR15042704 SRR15042705 SRR15042706 SRR15042707 SRR15042708
## ENSG00000000003    16.91709    6.699784   10.166800    5.484069    8.091160
## ENSG00000000005     0.00000    1.000000    0.000000    0.000000    1.000000
## ENSG00000000419   113.49260   51.035260   90.765940   17.384529   71.309920
## ENSG00000000457    58.57286   27.601510   43.664500    6.136820   28.670911
## ENSG00000000460    72.95979   19.233400   61.080660   12.899640   32.496790
## ENSG00000000938    85.99998  107.000040    8.999998    4.999999    1.999997
##                 SRR15042709 SRR15042712 SRR15042713 SRR15042714 SRR15042715
## ENSG00000000003    11.20247     3.48790    3.979877    12.26533    3.857950
## ENSG00000000005     0.00000     0.00000    0.000000     0.00000    0.000000
## ENSG00000000419    38.09333    85.88733  136.616960    55.81173    9.480490
## ENSG00000000457    49.87515    58.74561   24.360180    33.95916    5.110010
## ENSG00000000460    27.65710    41.24136   73.622270    21.13112    9.276386
## ENSG00000000938    14.00000    14.99996   16.999984   147.00005   11.000000
##                 SRR15042716 SRR15042717
## ENSG00000000003    1.204423    1.841854
## ENSG00000000005    0.000000    0.000000
## ENSG00000000419   12.664954   23.697475
## ENSG00000000457    8.742250   15.108520
## ENSG00000000460   18.097300   12.776200
## ENSG00000000938    8.000000   38.999944
```

```
dim(tcell_txi$counts)
```

```
## [1] 62703    12
```

3 Annotation

Sample annotation. Our interest mainly lies in the difference between expression profiles based on severity of the disease.

```
# Load annotation data and manipulate to fit the analysis
tcell_annotation <- read.table('tcell_annotation.txt', header=T, sep=',')
tcell_annotation <- dplyr::filter(tcell_annotation, Run %in% tcell_samples)
tcell_annotation$Severity[tcell_annotation$
                          Severity=="hospitalized moderate to severe"] <- 'Infected'
print(tcell_annotation[c("Run", "Severity", "Organism", "Tissue", "Cell_type",
                         "AvgSpotLen","Instrument", "LibraryLayout")])
```

```
##            Run Severity     Organism Tissue     Cell_type AvgSpotLen
## 1  SRR15042704  Healthy Homo sapiens  Blood CD4 TReg Cell         76
## 2  SRR15042705  Healthy Homo sapiens  Blood CD4 TReg Cell         76
## 3  SRR15042706  Healthy Homo sapiens  Blood CD4 TReg Cell         76
## 4  SRR15042707  Healthy Homo sapiens  Blood CD4 TReg Cell         76
```

```
## 5  SRR15042708  Healthy Homo sapiens  Blood CD4 TReg Cell         76
## 6  SRR15042709  Healthy Homo sapiens  Blood CD4 TReg Cell         76
## 7  SRR15042712 Infected Homo sapiens  Blood CD4 TReg Cell         76
## 8  SRR15042713 Infected Homo sapiens  Blood CD4 TReg Cell         76
## 9  SRR15042714 Infected Homo sapiens  Blood CD4 TReg Cell         76
## 10 SRR15042715 Infected Homo sapiens  Blood CD4 TReg Cell         76
## 11 SRR15042716 Infected Homo sapiens  Blood CD4 TReg Cell         76
## 12 SRR15042717 Infected Homo sapiens  Blood CD4 TReg Cell         76
##     Instrument LibraryLayout
## 1  NextSeq 500        PAIRED
## 2  NextSeq 500        PAIRED
## 3  NextSeq 500        PAIRED
## 4  NextSeq 500        PAIRED
## 5  NextSeq 500        PAIRED
## 6  NextSeq 500        PAIRED
## 7  NextSeq 500        PAIRED
## 8  NextSeq 500        PAIRED
## 9  NextSeq 500        PAIRED
## 10 NextSeq 500        PAIRED
## 11 NextSeq 500        PAIRED
## 12 NextSeq 500        PAIRED
```

```
# Annotation for design matrix
tcell_condition <- as.factor(tcell_annotation$Severity)
```

4 Preprocessing

4.1 Check for duplicate rows

```
sum(duplicated(rownames(tcell_txi$counts)))
```

```
## [1] 0
```

As no duplicates were found, we move on with the analysis.

5 Statistical Analysis

5.1 Normalization

```
# MAke edgeR-comapatible tpm values
tcell_cts <- tcell_txi$counts
tcell_normMat <- tcell_txi$length
# Calculate scaling factors
tcell_normMat <- tcell_normMat/exp(rowMeans(log(tcell_normMat)))
tcell_normCts <- tcell_cts/tcell_normMat
# Calculate effective library sizes
tcell_eff.lib <- calcNormFactors(tcell_normCts) * colSums(tcell_normCts)
```

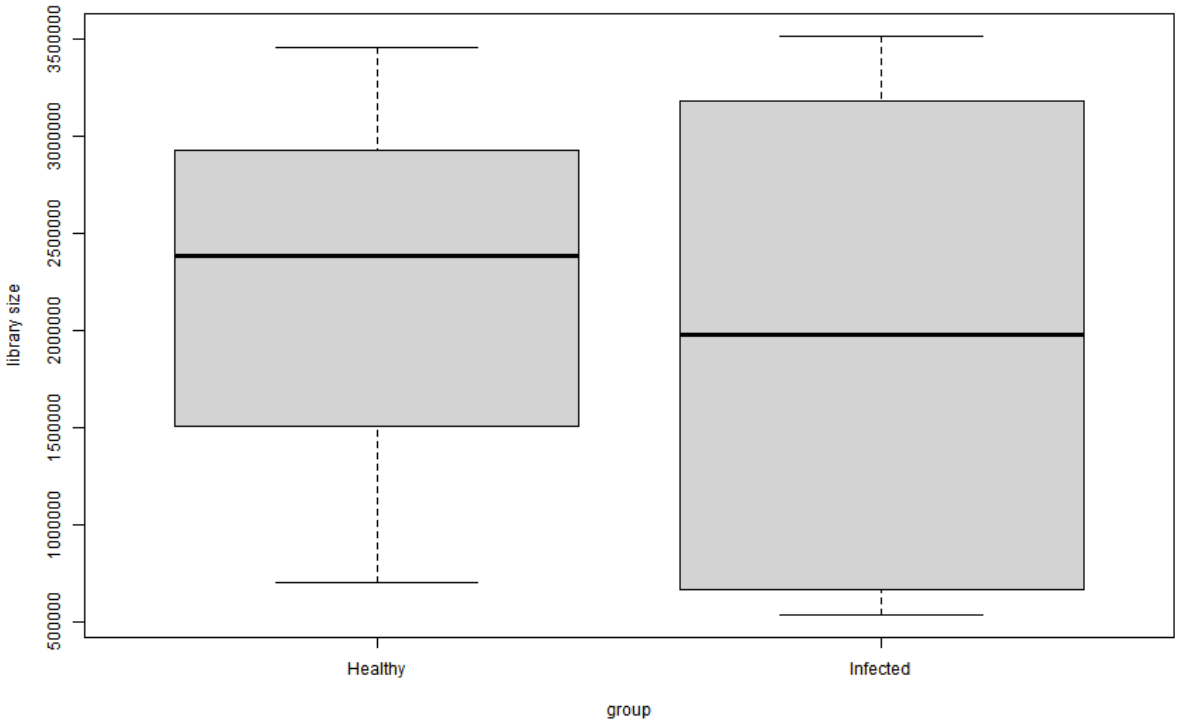
```
# Combine effective library sizes with length factors
tcell_normMat <- sweep(tcell_normMat, 2, tcell_eff.lib, "*")
# Calculate offsets
tcell_normMat <- log(tcell_normMat)
```

5.2 Library sizes

```
# Effective library sizes
tcell_eff.lib
```

```
## SRR15042704 SRR15042705 SRR15042706 SRR15042707 SRR15042708 SRR15042709
##   3451806.5   1506567.7   2870599.7    699818.7   2926058.9   1892435.8
## SRR15042712 SRR15042713 SRR15042714 SRR15042715 SRR15042716 SRR15042717
##   3093888.0   3511517.2   3174707.3    538149.3    669100.7    860316.4
```

```
boxplot(tcell_eff.lib~tcell_condition,xlab="group",ylab="library size")
```




```
jpeg("images/tcell_libsize.jpg")
boxplot(tcell_eff.lib~tcell_condition,xlab="group",ylab="library size")
dev.off()
```

```
## png
##   2
```

The boxplot shows that there is no significant difference in library effect sizes between healthy donors and COVID-19 patients.

5.2.1 Wilcox test

```
# Wilcox rank sum test for effective library sizes
wilcox.test(tcell_eff.lib~tcell_condition)
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  tcell_eff.lib by tcell_condition
## W = 19, p-value = 0.9372
## alternative hypothesis: true location shift is not equal to 0
```

The wilcox rank sum test gives a p-value of 0.9372, which is above 0.05. Therefore, we cannot reject the null hypothesis, which states that the median of library effect sizes are equal.

5.3 DGEList object

```
tcell_y <- DGEList(tcell_cts)
tcell_y <- scaleOffset(tcell_y, tcell_normMat)
```

5.4 Filtering on counts

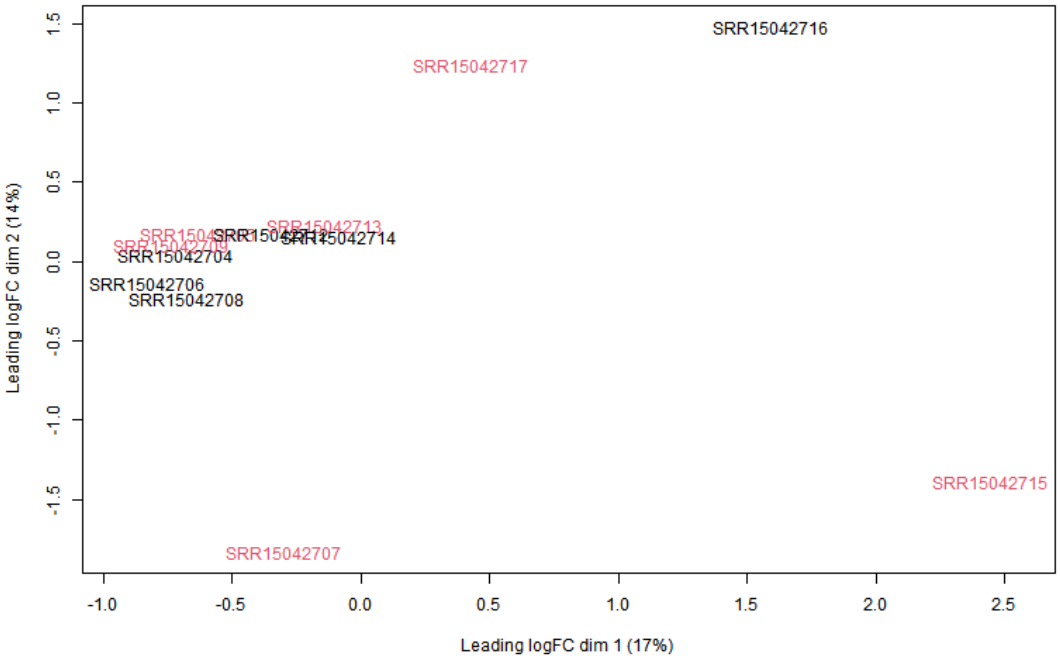
```
tcell_cutoff <- 3/(mean(tcell_y$samples$lib.size)/1000000)
tcell_keep <- rowSums(edgeR::cpm(tcell_y)>tcell_cutoff) >= 3
tcell_y <- tcell_y[tcell_keep, ,keep.lib.sizes=FALSE]
summary(tcell_keep)
```

```
##    Mode   FALSE    TRUE
## logical   40298   22405
```

We remove 40298 genes with low counts,

5.5 MDS plot

```
par(mar=c(6,6,6,6))
plotMDS.DGEList(tcell_y,col=as.double(sort(unique(tcell_condition))))
par(xpd=T)
legend(par("usr")[2]*-1.4,par("usr")[4]*1,sort(unique(tcell_condition)),
       pch=c(16),col=as.double(sort(unique(tcell_condition))))
```



There does not seem to be a clear separation of log fold changes between healthy donors and COVID-19 patients. Furthermore, a lot of the samples, regardless of infection, seem to group together at one region. This could be due to some confounder, but as the authors of the dataset did not enclose any patient information, it cannot be determined.

6 Differential Expression Analysis

6.1 Make design matrix

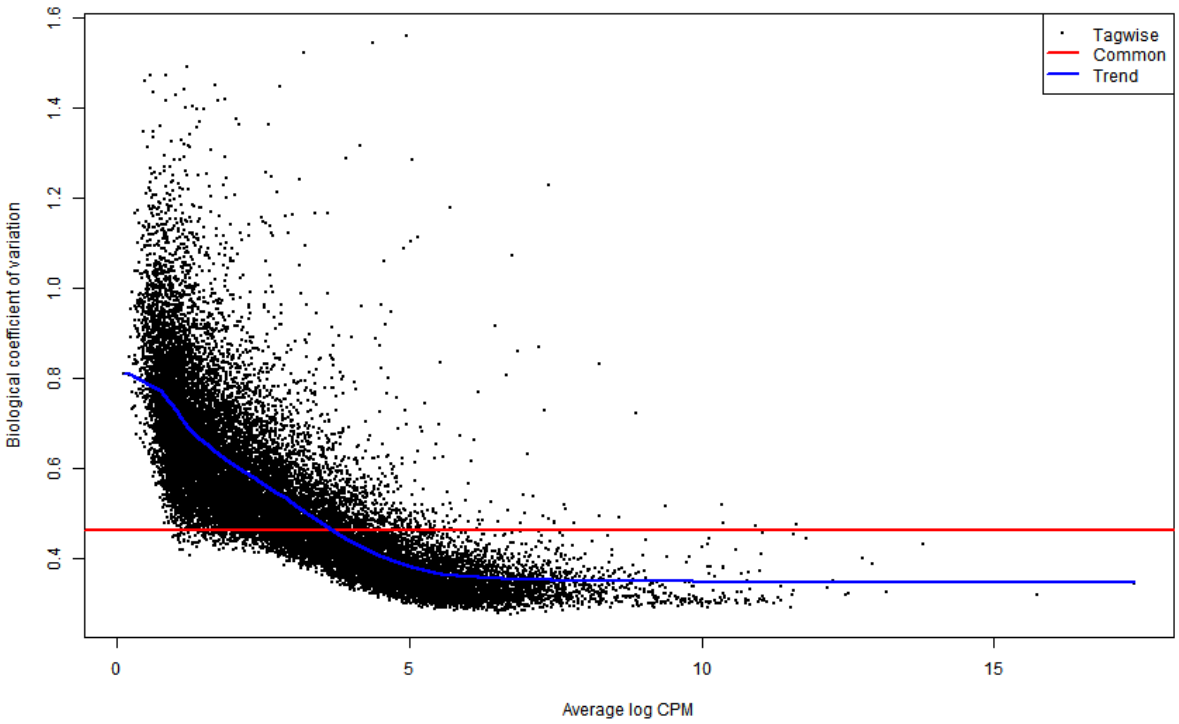
```
tcell_design <- model.matrix(~tcell_condition)
rownames(tcell_design) <- colnames(tcell_y)
tcell_design
```

```
##             (Intercept) tcell_conditionInfected
## SRR15042704           1                       0
## SRR15042705           1                       0
## SRR15042706           1                       0
## SRR15042707           1                       0
## SRR15042708           1                       0
## SRR15042709           1                       0
## SRR15042712           1                       1
## SRR15042713           1                       1
```

```
## SRR15042714           1                       1
## SRR15042715           1                       1
## SRR15042716           1                       1
## SRR15042717           1                       1
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$tcell_condition
## [1] "contr.treatment"
```

6.2 Calculate dispersion

```
tcell_y <- estimateDisp(tcell_y,tcell_design)
plotBCV(tcell_y)
```



The dispersion contains high variation (up to around 1.4 - above 0.4), but this applies to low log CPM values below 4. Therefore, the biological variation in the data is acceptable.

6.3 Quasi-likelihood test

```
# Make contrast for quasi-likelihood test
tcell_contrast <- makeContrast("tcell_conditionInfected=0",
                               parameterNames=colnames(tcell_design))
tcell_fit <- glmQLFit(tcell_y,tcell_design)
tcell_qlf <- glmQLFTest(tcell_fit, contrast=tcell_contrast)
tcell_toptable <- topTags(tcell_qlf,n=nrow(tcell_qlf$table))$table
head(tcell_toptable)
```

```
##                     logFC   logCPM        F       PValue         FDR
## ENSG00000081059 -1.932328 7.149506 51.79298 8.007115e-08 0.001515181
## ENSG00000102760 -2.372796 4.763248 47.32116 1.818264e-07 0.001515181
## ENSG00000130518 -2.147297 7.214625 46.74359 2.028807e-07 0.001515181
## ENSG00000229807 -6.279352 8.258060 44.25579 3.285729e-07 0.001840419
## ENSG00000235532 -2.202009 4.833268 41.56866 5.640737e-07 0.002527614
## ENSG00000175063  2.913994 4.093375 38.06415 1.180203e-06 0.003801939
```

```
sum(tcell_toptable$FDR < 0.05)
```

```
## [1] 95
```

There are 95 differentially expressed genes with FDR below 0.05.

6.4 Addition of gene symbols

6.4.1 Get gene symbols

```
data_sorted <- data[sort(data$ensembl_gene_id,index.return=T)$ix,]
data_sorted <- data_sorted[!duplicated(data_sorted$ensembl_gene_id),]
```

6.4.2 Add gene symbols to `toptable`

```
tcell_toptable <- cbind(rownames(tcell_toptable),tcell_toptable)
colnames(tcell_toptable)[1] <- "Ensembl_gene_id"
tcell_toptable_sorted <- tcell_toptable[sort(tcell_toptable$Ensembl_gene_id,
                                             index.return=T)$ix,]
tcell_data_sorted <- data_sorted[data_sorted$ensembl_gene_id%in%
                                 tcell_toptable_sorted$Ensembl_gene_id,]
dim(tcell_toptable)
```

```
## [1] 22405     6
```

```
dim(tcell_data_sorted)
```

```
## [1] 22405     3
```

```
tcell_toptable_sorted$Gene_symbol <- tcell_data_sorted$external_gene_name
head(tcell_toptable_sorted)
```

```
##                 Ensembl_gene_id        logFC   logCPM            F    PValue
## ENSG00000000003 ENSG00000000003 -0.862362917 2.232251 2.3447335333 0.1369719
## ENSG00000000419 ENSG00000000419 -0.040096458 4.810875 0.0171131198 0.8968590
## ENSG00000000457 ENSG00000000457 -0.349058569 3.988586 0.7389164728 0.3973366
## ENSG00000000460 ENSG00000000460 -0.049816762 4.123408 0.0208371460 0.8862615
## ENSG00000000938 ENSG00000000938  0.261968145 4.494728 0.1784330292 0.6759621
## ENSG00000000971 ENSG00000000971 -0.007640368 3.747487 0.0003586344 0.9850257
##                       FDR Gene_symbol
## ENSG00000000003 0.6148943      TSPAN6
## ENSG00000000419 0.9813038        DPM1
## ENSG00000000457 0.8109466       SCYL3
## ENSG00000000460 0.9796154    C1orf112
## ENSG00000000938 0.9227399         FGR
## ENSG00000000971 0.9977504         CFH
```

6.4.3 Resort data

```
tcell_toptable <- tcell_toptable_sorted[sort(
                               tcell_toptable_sorted$PValue,index.return=T)$ix,]
```

6.4.4 Explore and save results

```
head(tcell_toptable[,c(1,7,2,5,6)],10)
```

```
##                 Ensembl_gene_id Gene_symbol     logFC       PValue         FDR
## ENSG00000081059 ENSG00000081059        TCF7 -1.932328 8.007115e-08 0.001515181
## ENSG00000102760 ENSG00000102760        RGCC -2.372796 1.818264e-07 0.001515181
## ENSG00000130518 ENSG00000130518        IQCN -2.147297 2.028807e-07 0.001515181
## ENSG00000229807 ENSG00000229807        XIST -6.279352 3.285729e-07 0.001840419
## ENSG00000235532 ENSG00000235532   LINC00402 -2.202009 5.640737e-07 0.002527614
## ENSG00000175063 ENSG00000175063       UBE2C  2.913994 1.180203e-06 0.003801939
## ENSG00000145632 ENSG00000145632        PLK2 -3.829977 1.187841e-06 0.003801939
## ENSG00000163736 ENSG00000163736        PPBP  5.325215 1.415409e-06 0.003964030
## ENSG00000189159 ENSG00000189159        JPT1  1.574384 1.646996e-06 0.004100106
## ENSG00000167900 ENSG00000167900         TK1  2.507733 1.906440e-06 0.004271379
```

```
write.table(tcell_toptable,file="result/tcell_toptable.txt",col.names=T,
            row.names=F,sep="\t",quote=F)
tcell_toptable_sign <- tcell_toptable[tcell_toptable$FDR<0.05,]
dim(tcell_toptable_sign)
```

```
## [1] 95  7
```

```
head(tcell_toptable_sign,10)
```

```
##                 Ensembl_gene_id     logFC   logCPM        F       PValue
## ENSG00000081059 ENSG00000081059 -1.932328 7.149506 51.79298 8.007115e-08
## ENSG00000102760 ENSG00000102760 -2.372796 4.763248 47.32116 1.818264e-07
## ENSG00000130518 ENSG00000130518 -2.147297 7.214625 46.74359 2.028807e-07
## ENSG00000229807 ENSG00000229807 -6.279352 8.258060 44.25579 3.285729e-07
## ENSG00000235532 ENSG00000235532 -2.202009 4.833268 41.56866 5.640737e-07
## ENSG00000175063 ENSG00000175063  2.913994 4.093375 38.06415 1.180203e-06
## ENSG00000145632 ENSG00000145632 -3.829977 3.577027 38.03435 1.187841e-06
## ENSG00000163736 ENSG00000163736  5.325215 4.651867 37.22994 1.415409e-06
## ENSG00000189159 ENSG00000189159  1.574384 6.060132 36.54273 1.646996e-06
## ENSG00000167900 ENSG00000167900  2.507733 5.129287 35.88648 1.906440e-06
##                         FDR Gene_symbol
## ENSG00000081059 0.001515181        TCF7
## ENSG00000102760 0.001515181        RGCC
## ENSG00000130518 0.001515181        IQCN
## ENSG00000229807 0.001840419        XIST
## ENSG00000235532 0.002527614   LINC00402
## ENSG00000175063 0.003801939       UBE2C
## ENSG00000145632 0.003801939        PLK2
## ENSG00000163736 0.003964030        PPBP
## ENSG00000189159 0.004100106        JPT1
## ENSG00000167900 0.004271379         TK1
```

```
dim(tcell_toptable_sign[tcell_toptable_sign$logFC>0,])
```

```
## [1] 55  7
```

```
dim(tcell_toptable_sign[tcell_toptable_sign$logFC<0,])
```

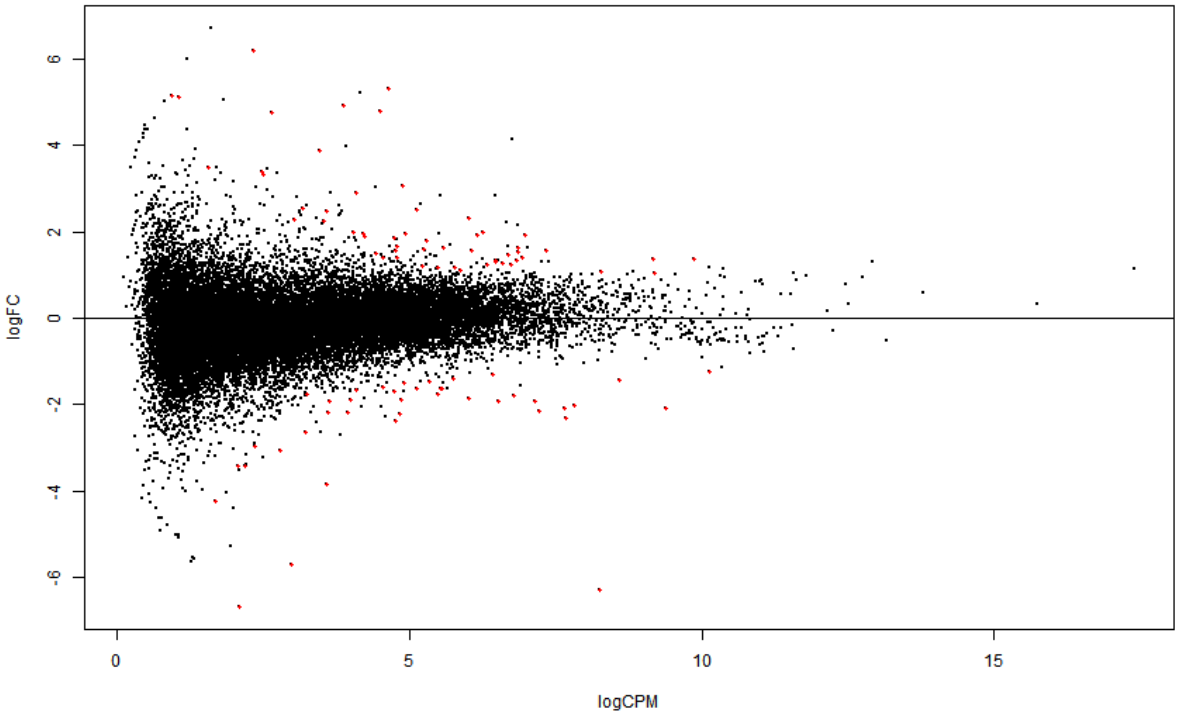
```
## [1] 40  7
```

Around 0.4% (95 out of 22405) of genes were differentially expressed at FDR below 0.05.

6.5 Plots

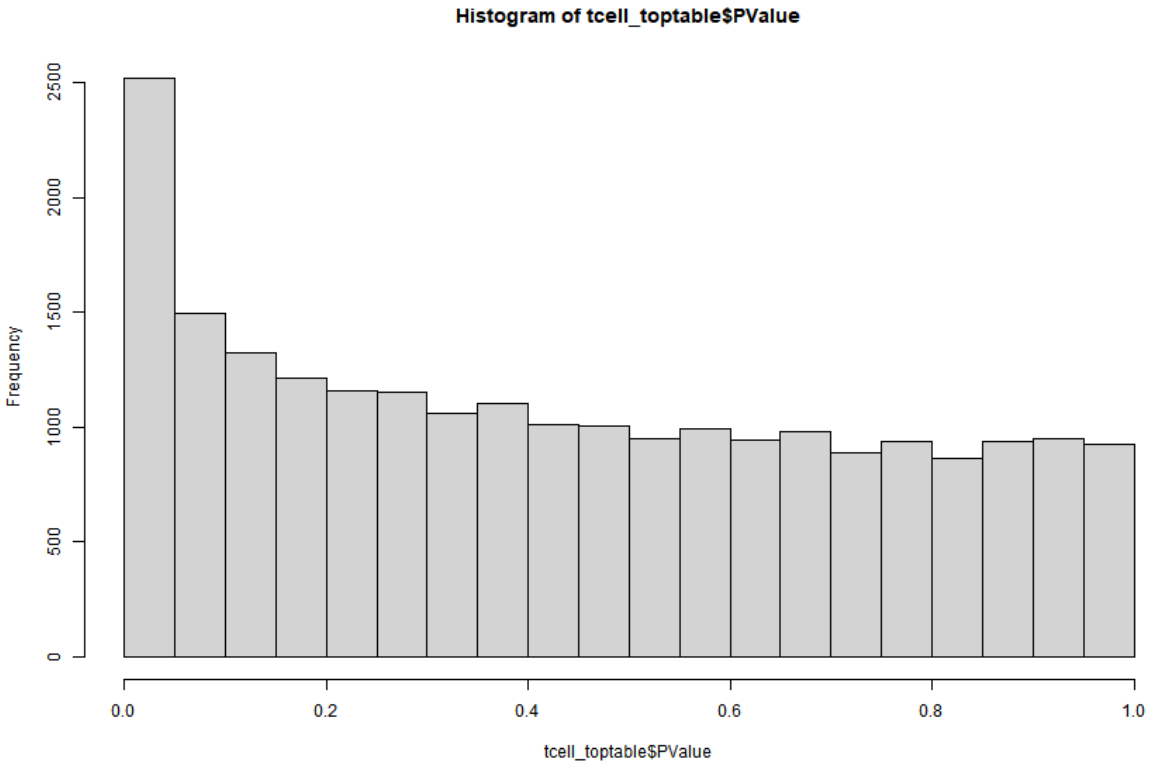
6.5.1 MA plot

```
with(tcell_toptable,plot(logCPM,logFC,pch=16,cex=0.2))
# MAplot: all data points
with(tcell_toptable,points(logCPM[FDR<0.05],logFC[FDR<0.05],pch=16,col="red",cex=0.6))
# MA-plot: significant loci
abline(0,0)
```



6.5.2 P-value distribution

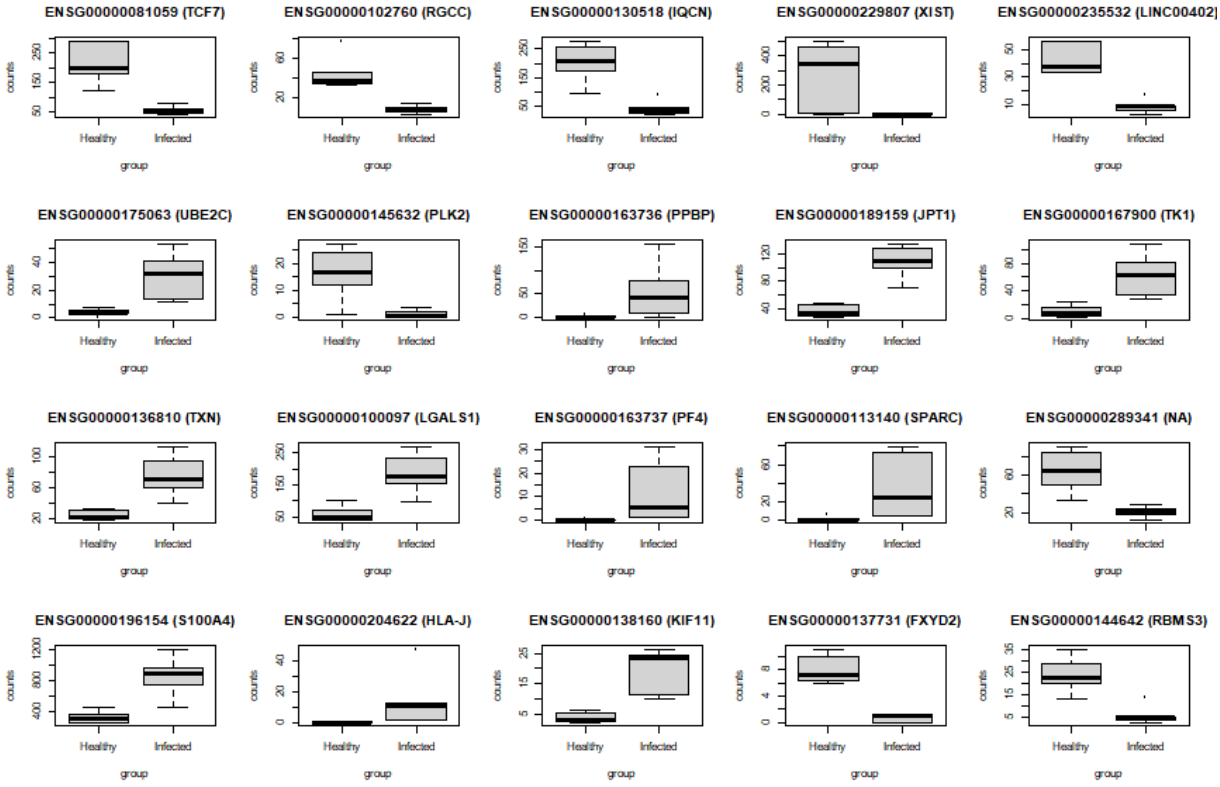
```
hist(tcell_toptable$PValue)
```



The p-value distribution is not uniformly distributed.

6.5.3 Boxplots of top 20 loci

```
par(mfrow=c(4,5))
tcell_counts_k <- tcell_txi$counts[tcell_keep,]
for (i in 1:20){
  tcell_counts_part <- as.numeric(edgeR::cpm(tcell_y)[rownames(tcell_counts_k)
                                                      ==rownames(tcell_toptable)[i],])
  tcell_boxplot <- data.frame(counts=tcell_counts_part,group=tcell_condition)
  if (tcell_toptable$Gene_symbol[i]!=""){
    boxplot(counts~group,tcell_boxplot,main=paste(rownames(tcell_toptable)[i],
                                                  " (",
                                                  tcell_toptable$Gene_symbol[i],
                                                  ")",sep=""))
  } else {
    boxplot(counts~group,tcell_boxplot,main=paste(rownames(tcell_toptable)[i],
                                                  " (NA)", sep=""))
  }
}
```



There is a clear separation between the two groups.

Some are very clearly divided, such as gene JPT1, but there are also some that does seem slightly problematic. There are some boxplots that show high variation of counts. For example, the healthy counts of gene PLK2 has very long whiskers that reach to almost the median of the infected counts. Another example is the boxplot of gene XIST. The box of the healthy counts span from near 0 to over 400. This has large variation is in the interquantile range.

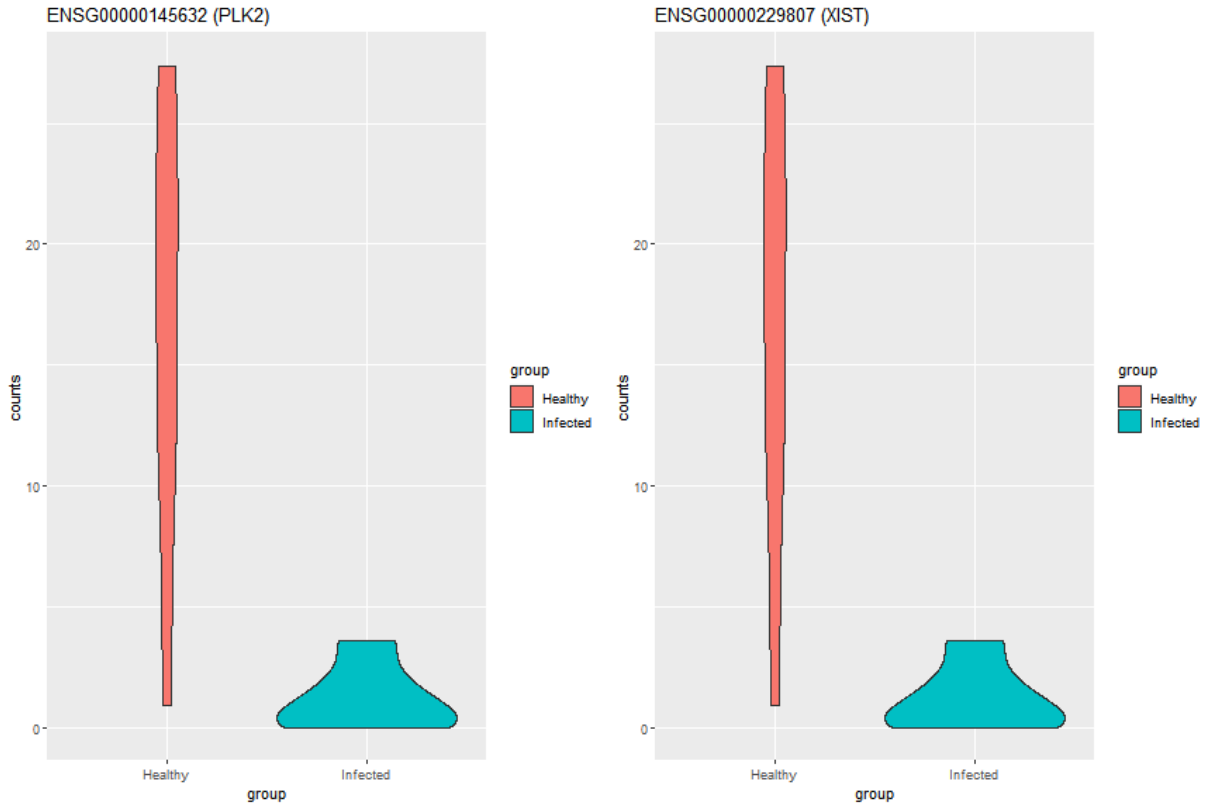
While the results may be dubious, the sample size is small, which may give this result. Furthermore, even with high variation, the difference between the two groups are still quite visible.

To look into the two examples of genes where the boxplots were dubious, we plot them into violin plots.

6.5.4 Violin plots of top 20 loci

```
# Violin plot of gene PLK2
tcell_counts_plk2 <- as.numeric(edgeR::cpm(tcell_y)
                                [rownames(tcell_toptable[tcell_toptable$Gene_symbol
                                                         =="PLK2",]),])
tcell_boxplot_plk2 <- data.frame(counts=tcell_counts_plk2,group=tcell_condition)
violin_plk2 <- ggplot(tcell_boxplot_plk2, aes(x=group, y=counts, fill=group)) +
               ggtitle(paste(rownames(tcell_toptable[tcell_toptable$Gene_symbol
                                                     =="PLK2",]),
                             " (", "PLK2", ")",sep="")) +
               geom_violin()
# Violin plot of gene XIST
```

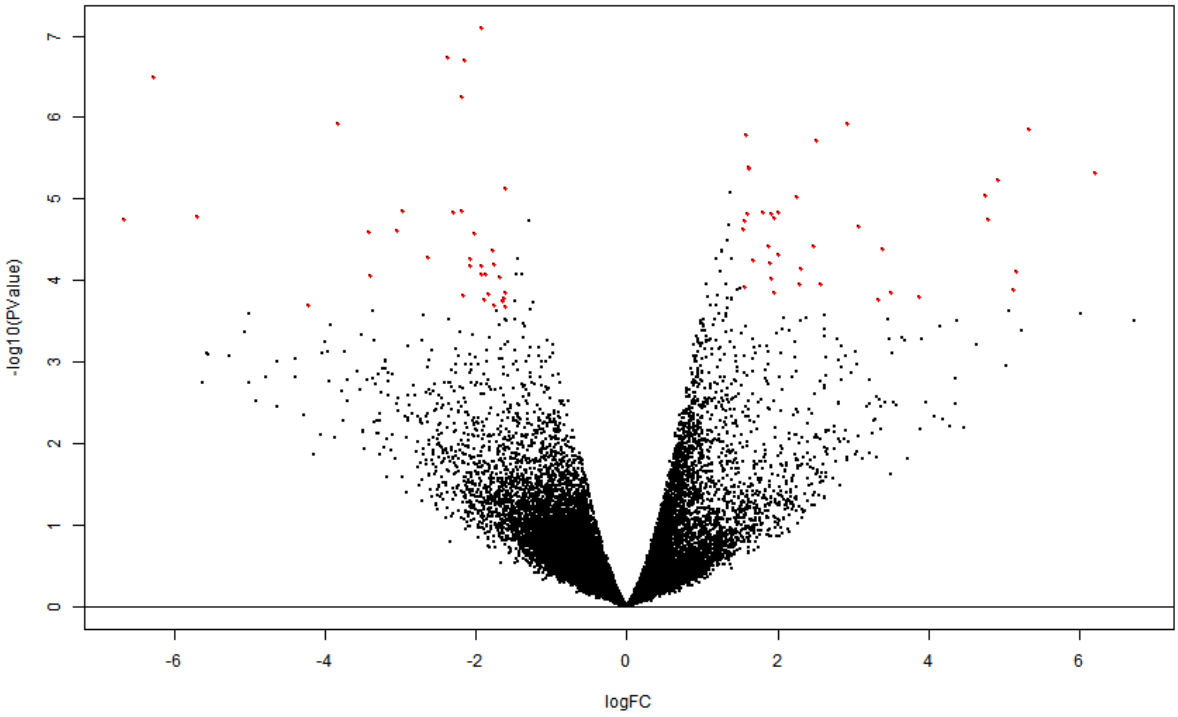
```
tcell_counts_xist <- as.numeric(edgeR::cpm(tcell_y)
                                [rownames(tcell_toptable[tcell_toptable$Gene_symbol
                                                         =="XIST",]),])
tcell_boxplot_xist <- data.frame(counts=tcell_counts_xist,group=tcell_condition)
violin_xist <- ggplot(tcell_boxplot_plk2, aes(x=group, y=counts, fill=group)) +
               ggtitle(paste(rownames(tcell_toptable[tcell_toptable$Gene_symbol
                                                     =="XIST",]),
                             " (", "XIST", ")",sep="")) +
               geom_violin()
grid.arrange(violin_plk2, violin_xist, nrow = 1)
```



The violin plot shows similar variation as seen in the boxplots. And it shows that there is high variation and small density, shown in gene XIST. This is most likely due to the small sample size.

6.5.5 Volcano plot

```
with(tcell_toptable,plot(logFC,-log10(PValue),pch=16,cex=0.2))
with(tcell_toptable,points(logFC[FDR<0.05 & abs(logFC)>1.5],
                           -log10(PValue)[FDR<0.05 & abs(logFC)>1.5],
                           pch=16,col="red",cex=0.6))
abline(0,0)
```



The volcano plot shows similar result as the MA plot. There are almost similar amount of up- and down-regulated genes.

7 Gene set analysis

Gene set analysis is performed to get more insight into the results. Specifically, we make use of overrepresentation analysis (ORA).

7.1 Annotation

For annotation, the `data` from Section 1, human annotation, is used. Duplicated ensembl gene IDs are removed and the data is saved in `.txt` file.

```
if(!file.exists("RefGenes.txt")){
  refgenes <- data[!duplicated(data$ensembl_gene_id),]
  refgenes <- refgenes[c("ensembl_gene_id")]
  write.table(refgenes,file="RefGenes.txt", col.names=c("Ensembl_gene_id"),
              row.names=F, quote=F, sep='\t')
}
```

7.2 Gene set analysis using custom gene sets

7.2.1 Make gmt file for WebGestaltR

The file format gmt should contain the gene sets.

```
if (!file.exists("geneset_full.gmt")){
  geneset <- read.csv("geneset_full.csv", header=F)
  write.table(geneset, file="geneset_full.gmt",col.names=T,row.names=F,sep="\t",
              quote=F,na="")
}
```

7.2.2 Run WebGestaltR

```
tcell_enrichResult_full <- WebGestaltR(enrichMethod="ORA",
                              organism="hsapiens",
                              enrichDatabase="others",
                              enrichDatabaseFile="geneset_full.gmt",
                              enrichDatabaseType="genesymbol",
                              interestGeneFile="result/tcell_toptable.txt",
                              interestGeneType="ensembl_gene_id",
                              referenceGeneFile="RefGenes.txt",
                              referenceGeneType="ensembl_gene_id",
                              sigMethod="fdr",
                              fdrThr=0.05,
                              outputDirectory='result',
                              projectName = 'ORA_tcell',
                              is.ouput=TRUE)
```

7.3 Gene set analysis using KEGG pathway

```
tcell_enrichResult_kegg <- WebGestaltR(enrichMethod="ORA",
                              organism="hsapiens",
                              enrichDatabase="pathway_KEGG",
                              interestGeneFile="result/tcell_toptable.txt",
                              interestGeneType="ensembl_gene_id",
                              referenceGeneFile="RefGenes.txt",
                              referenceGeneType="ensembl_gene_id",
                              sigMethod="fdr",
                              fdrThr=0.05,
                              outputDirectory='result',
                              projectName = 'ORA_tcell_kegg',
                              is.ouput=TRUE)
```

7.3.1 Genes found from gene set analysis

```
#overlap_genes <- unique(unlist(strsplit(tcel_enrichResult_kegg$userID, split=';')))
#top_genesets <- tcell_enrichResult_kegg[tcell_enrichResult_kegg$description %in% c("Endocytosis", "Apo
#top_overlaps <- unique(unlist(strsplit(top_genesets$userID, split=';')))
```

B.3 Results

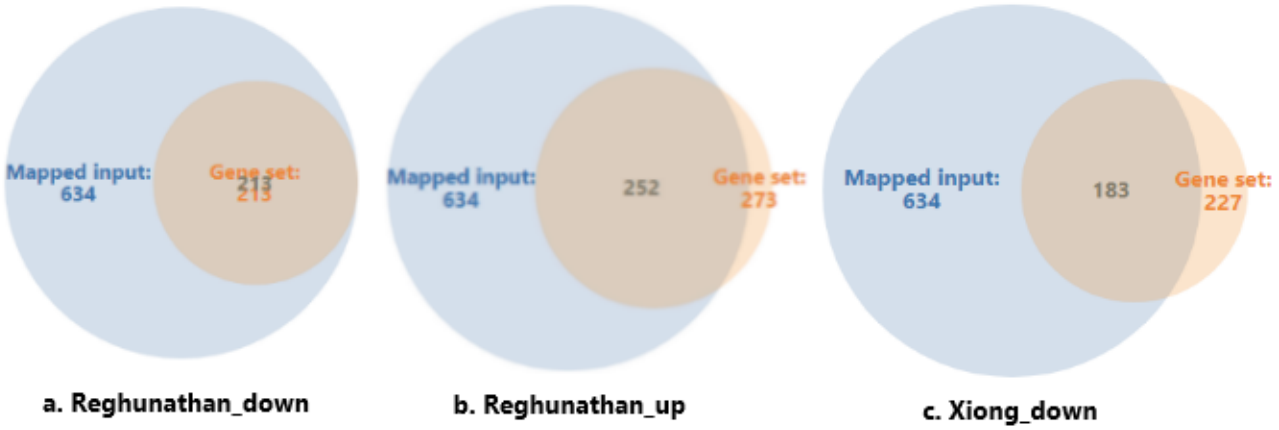


Figure S5: Overlaps between the mapped genes from monocyte RNA-seq data and the (a) downregulated genes from Reghunathan et al., (b) upregulated genes from Reghunathan et al., and (c) downregulated genes from Xiong et al.

Table S1: *Top 30 enrichment results of the T-cell RNA-seq data using KEGG pathway with a threshold at FDR below 0.05.*

Gene Set	Description	Size	Expect	Ratio	*P-value*	*FDR*
hsa03040	Spliceosome	134	99.759	1.3132	8.0380e-14	2.5320e-11
hsa04141	Protein processing in endoplasmic reticulum	165	122.84	1.2700	7.6110e-12	8.0424e-10
hsa04144	Endocytosis	244	181.65	1.2276	7.6594e-12	8.0424e-10
hsa05016	Huntington disease	192	142.94	1.2453	4.2051e-11	3.2082e-9
hsa04714	Thermogenesis	229	170.48	1.2259	5.0924e-11	3.2082e-9
hsa04142	Lysosome	123	91.570	1.2886	1.5712e-10	8.2487e-9
hsa03013	RNA transport	171	127.30	1.2411	1.1138e-9	5.0120e-8
hsa04140	Autophagy	128	95.292	1.2698	1.9729e-9	7.7682e-8
hsa04110	Cell cycle	124	92.314	1.2674	5.2436e-9	1.8353e-7
hsa04120	Ubiquitin mediated proteolysis	137	101.99	1.2550	6.4176e-9	2.0216e-7
hsa04932	Non-alcoholic fatty liver disease (NAFLD)	149	110.93	1.2441	8.5459e-9	2.4472e-7
hsa05203	Viral carcinogenesis	201	149.64	1.2096	1.6334e-8	4.2877e-7
hsa04210	Apoptosis	136	101.25	1.2445	3.7070e-8	8.9825e-7
hsa00240	Pyrimidine metabolism	101	75.191	1.2767	4.3071e-8	9.6911e-7
hsa05010	Alzheimer disease	171	127.30	1.2176	5.9746e-8	0.0000012547
hsa04071	Sphingolipid signaling pathway	118	87.847	1.2522	1.1193e-7	0.0000022037
hsa04380	Osteoclast differentiation	128	95.292	1.2383	2.2121e-7	0.0000040988
hsa04660	T cell receptor signaling pathway	101	75.191	1.2634	2.4881e-7	0.0000043542
hsa05145	Toxoplasmosis	113	84.125	1.2481	3.5330e-7	0.0000058573
hsa03420	Nucleotide excision repair	47	34.990	1.3432	9.0142e-7	0.000014197
hsa00970	Aminoacyl-tRNA biosynthesis	44	32.757	1.3432	0.0000021985	0.000030348
hsa03460	Fanconi anemia pathway	54	40.201	1.3184	0.0000022159	0.000030348
hsa00230	Purine metabolism	174	129.54	1.1888	0.0000027973	0.000034501
hsa00562	Inositol phosphate metabolism	74	55.091	1.2706	0.0000057492	0.000056594
hsa05012	Parkinson disease	142	105.71	1.2013	0.0000059347	0.000056649
hsa05220	Chronic myeloid leukemia	76	56.580	1.2549	0.000018435	0.00016592
hsa03030	DNA replication	36	26.801	1.3432	0.000023645	0.00020690
hsa03050	Proteasome	45	33.501	1.3134	0.000027065	0.00023041
hsa03010	Ribosome	151	112.41	1.1742	0.000061440	0.00040763
hsa04659	Th17 cell differentiation	107	79.658	1.2052	0.000062115	0.00040763

C Monocyte RNA-seq Data Analysis

This section gives details on the data analysis of monocyte RNA-seq data and contains the analysis results.

C.1 Quality Control

FastQC was used for quality control and the FastQC results were combined via MultiQC. The following shows some of the relevant figures from quality control.

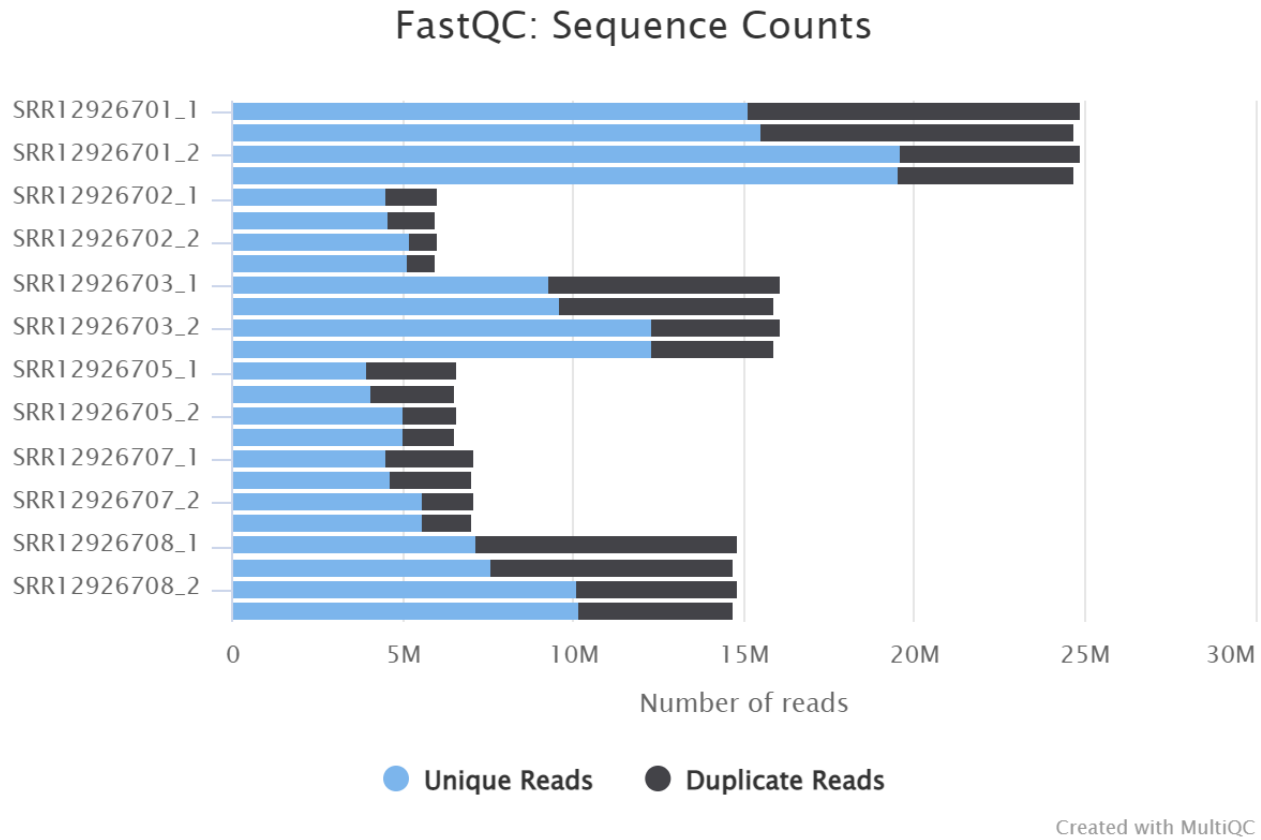


Figure S6: Sequence counts for each sample of monocyte RNA-seq data. Duplicate read counts are an estimate only. The extensions after the sample names indicate the forward read (*1*) and reverse read (*2*).

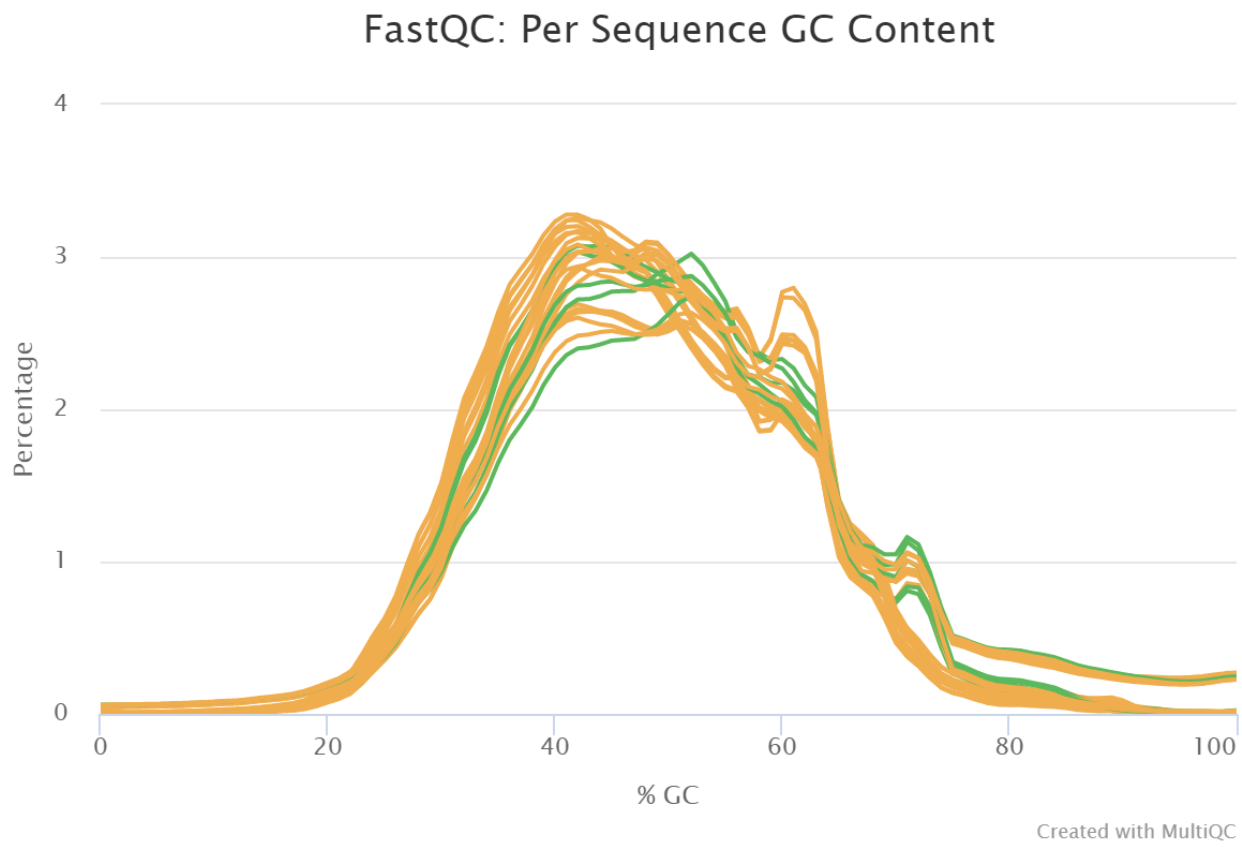


Figure S7: Per Sequence GC Content of monocyte RNA-seq data. The average GC content of reads.

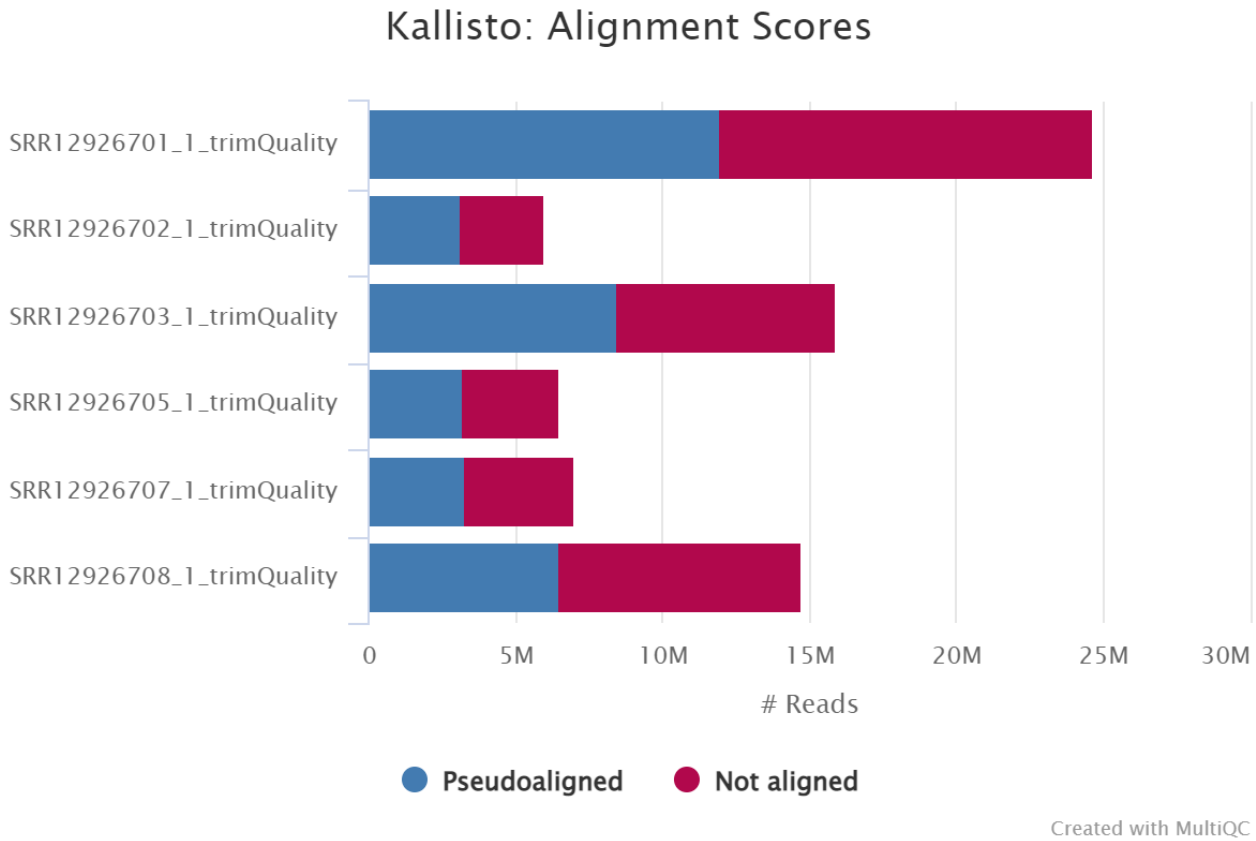


Figure S8: Kallisto alignment scores of monocyte RNA-seq data.

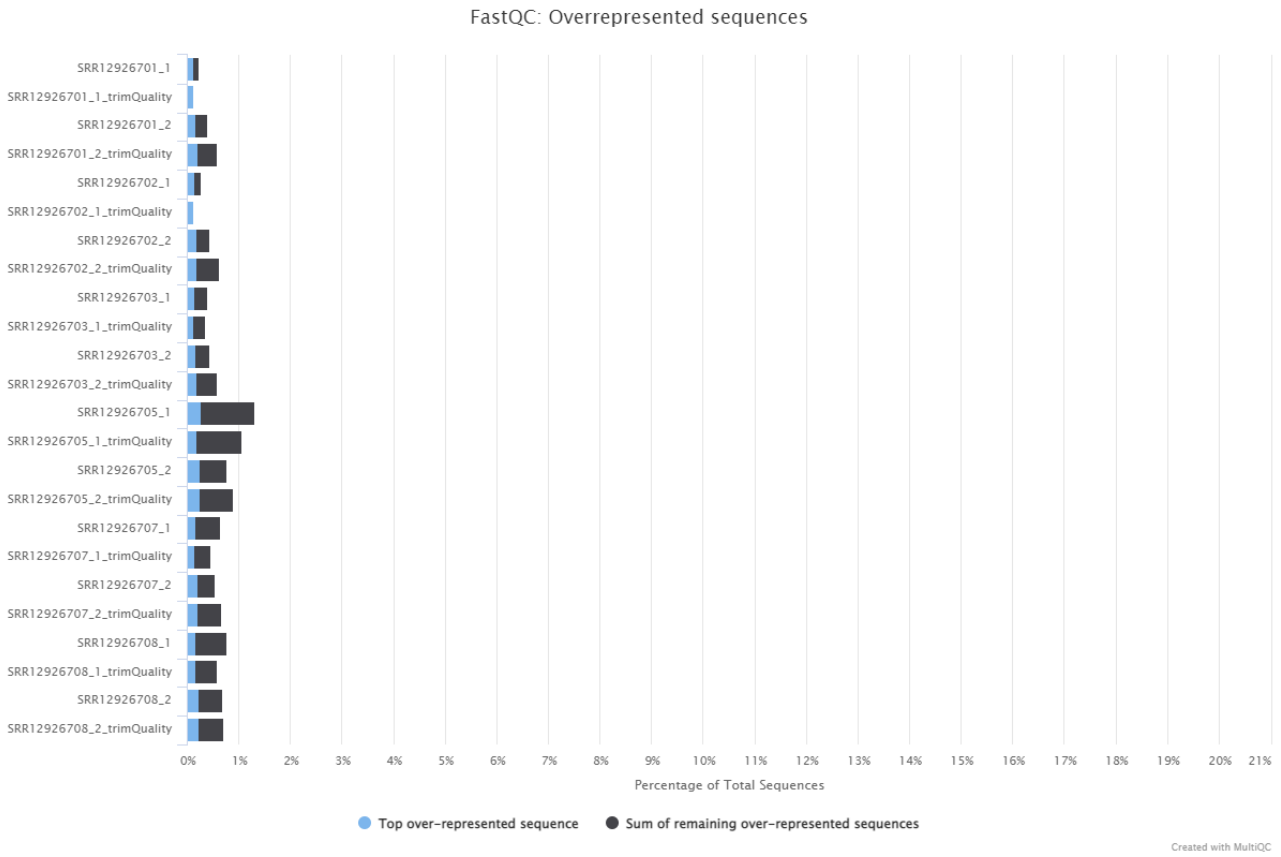


Figure S9: Overrepresented sequences of monocyte RNA-seq data. The total amount of overrepresented sequences found in each library. The extensions after the sample names indicate the forward read (*1*) and reverse read (*2*), and *trimQuality* refers to the samples trimmed for quality.

C.2 R Markdown of Data Analysis

The following is the R markdown file of the statistical analysis of monocyte RNA-seq Data.

1 Load libraries

```
suppressPackageStartupMessages({
  library(biomaRt)
  library(dplyr)
  library(tximport)
  library(edgeR)
  library(msqrob2)
  library(WebGestaltR)
})
```

2 PseudoCount table

2.1 Human annotation data

```
#annotation data
hs_annot <- useEnsembl(biomart = "ensembl", dataset = "hsapiens_gene_ensembl")

#attributes of annotation data
attributes <- listAttributes(hs_annot)

data <- getBM(attributes = c('ensembl_gene_id', 'ensembl_transcript_id',
                             'external_gene_name'),
              mart = hs_annot)

tx2geneGtf <- dplyr::select(data, ensembl_transcript_id, ensembl_gene_id)
tx2geneGtf <- dplyr::rename(tx2geneGtf, TXNAME = ensembl_transcript_id)
tx2geneGtf <- dplyr::rename(tx2geneGtf, GENEID = ensembl_gene_id)

head(tx2geneGtf)
```

```
##            TXNAME          GENEID
## 1 ENST00000387314 ENSG00000210049
## 2 ENST00000389680 ENSG00000211459
## 3 ENST00000387342 ENSG00000210077
## 4 ENST00000387347 ENSG00000210082
## 5 ENST00000386347 ENSG00000209082
## 6 ENST00000361390 ENSG00000198888
```

3 Load data

```
# Get file locations
blood_files <- list.files("data/RNAseq_blood/")
blood_files <- blood_files[grep("abundance.tsv",blood_files)]
blood_samples <- unlist(strsplit(blood_files,"_"))[c(1:length(blood_files))*2-1]
blood_files <- paste(rep("data/RNAseq_blood/",length(blood_files)),blood_files,sep="")
```

```
names(blood_files) <- blood_samples

# Load RNAseq data
blood_txi <- tximport(blood_files, type = "kallisto", tx2gene = tx2geneGtf)
head(blood_txi$counts)
```

```
##                 SRR12926701 SRR12926702 SRR12926703 SRR12926705 SRR12926707
## ENSG00000000003    12.72267    1.137453    2.918442    10.85208    5.335611
## ENSG00000000005     1.00000    0.000000    0.000000     0.00000    1.000000
## ENSG00000000419   395.97287   84.584600  288.772003    94.41785  118.155981
## ENSG00000000457   284.44905   78.052547  207.052600     9.02134   18.737600
## ENSG00000000460   107.68048   45.171150   76.691980    12.63891   15.670930
## ENSG00000000938  7076.99703 1927.000100 5595.004600  1841.00030 1508.000407
##                 SRR12926708
## ENSG00000000003    4.726439
## ENSG00000000005    1.000000
## ENSG00000000419  209.409720
## ENSG00000000457   31.207870
## ENSG00000000460   41.256740
## ENSG00000000938 3385.000000
```

```
dim(blood_txi$counts)
```

```
## [1] 62703     6
```

4 Annotation

Sample annotation. Our interest mainly lies in the difference between expression profiles based on severity of the disease.

```
# Load annotation data and manipulate to fit the analysis
blood_annotation <- read.table("monocyte_sra.txt", sep=',', header=T)
blood_annotation <- dplyr::filter(blood_annotation, Run %in% blood_samples)
blood_annotation["Condition"] <- t(t(t(blood_annotation["Condition"]) %>%
                                       strsplit(" ") %>% vapply("[","",1)))
print(blood_annotation[c("Run", "Sample.Name", "Condition", "AvgSpotLen",
                         "Organism", "source_name", "Instrument", "LibraryLayout")])
```

```
##           Run Sample.Name Condition AvgSpotLen     Organism
## 1 SRR12926701  GSM4872020   Healthy        150 Homo sapiens
## 2 SRR12926702  GSM4872021   Healthy        150 Homo sapiens
## 3 SRR12926703  GSM4872022   Healthy        150 Homo sapiens
## 4 SRR12926705  GSM4872024  COVID-19        150 Homo sapiens
## 5 SRR12926707  GSM4872026  COVID-19        150 Homo sapiens
## 6 SRR12926708  GSM4872027  COVID-19        150 Homo sapiens
##            source_name  Instrument LibraryLayout
## 1 Peripheral monocytes NextSeq 550        PAIRED
## 2 Peripheral monocytes NextSeq 550        PAIRED
## 3 Peripheral monocytes NextSeq 550        PAIRED
## 4 Peripheral monocytes NextSeq 550        PAIRED
## 5 Peripheral monocytes NextSeq 550        PAIRED
## 6 Peripheral monocytes NextSeq 550        PAIRED
```

```r
# Annotation for design matrix
blood_condition <- as.factor(t(blood_annotation["Condition"]))
#blood_patient <- as.factor(t(blood_annotation["Sample.Name"]))
```

5 Preprocessing

5.1 Check for duplicate rows

```r
sum(duplicated(rownames(blood_txi$counts)))
```

```
## [1] 0
```

As no duplicates were found, we move on with the analysis.

6 Statistical Analysis

6.1 Normalization

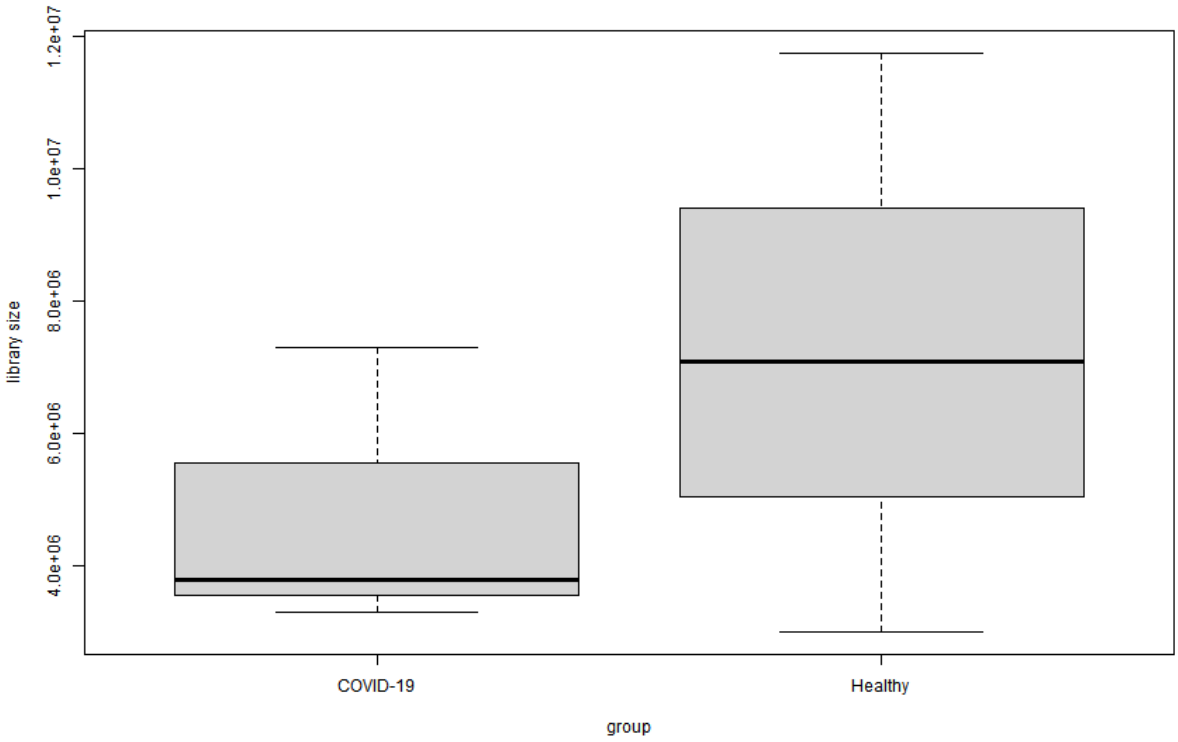
```r
# Make edgeR-comapatible tpm values
blood_cts <- blood_txi$counts
blood_normMat <- blood_txi$length
# Calculate scaling factors
blood_normMat <- blood_normMat/exp(rowMeans(log(blood_normMat)))
blood_normCts <- blood_cts/blood_normMat
# Calculate effective library sizes
blood_eff.lib <- calcNormFactors(blood_normCts) * colSums(blood_normCts)
# Combine effective library sizes with length factors
blood_normMat <- sweep(blood_normMat, 2, blood_eff.lib, "*")
# Calculate offsets
blood_normMat <- log(blood_normMat)
```

6.1.1 Library sizes

```r
# Effective library sizes
blood_eff.lib
```

```
## SRR12926701 SRR12926702 SRR12926703 SRR12926705 SRR12926707 SRR12926708
##    11748621     2998384     7076677     3292162     3784666     7301868
```

```r
boxplot(blood_eff.lib~blood_condition,xlab="group",ylab="library size")
```




```
# Wilcox rank sum test for effective library sizes
wilcox.test(blood_eff.lib~blood_condition)
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  blood_eff.lib by blood_condition
## W = 4, p-value = 1
## alternative hypothesis: true location shift is not equal to 0
```

Although the medians for library sizes between the two groups may seem different at first glance, there is quite a lot of variation.

Furhtermore, we see that the p-value of Wilcox rank sum test is 1, much larger than 0.05. Therefore, we cannot reject the null hypothesis that the medians are the same.

6.2 DGEList object

```
blood_y <- DGEList(blood_cts)
blood_y <- scaleOffset(blood_y, blood_normMat)
```

6.3 Filtering on counts

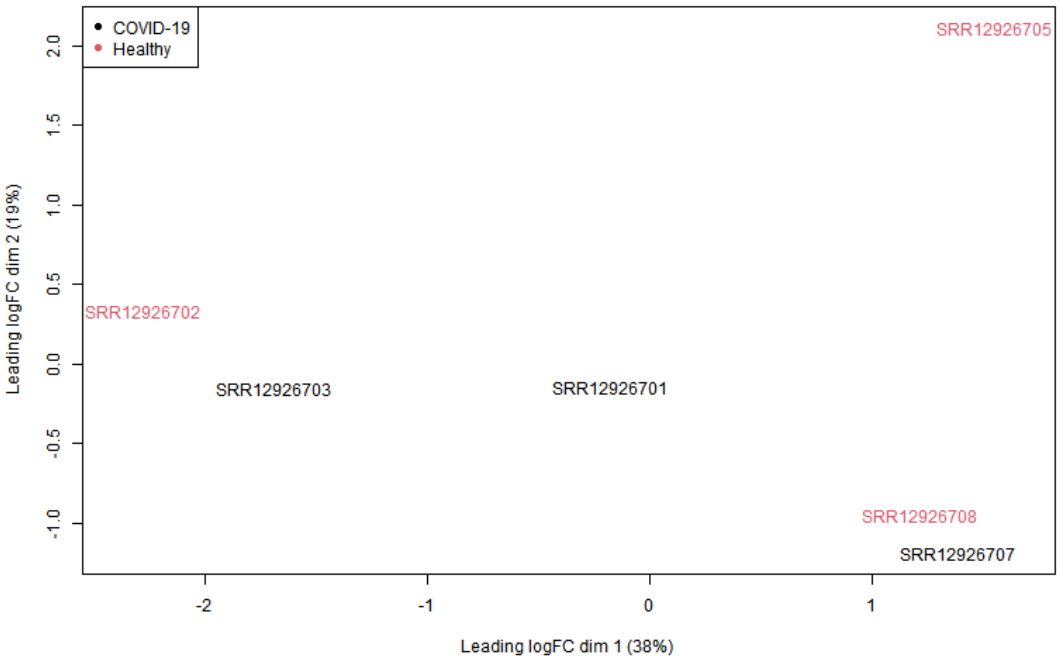
```
blood_cutoff <- 3/(mean(blood_y$samples$lib.size)/1000000)
blood_keep <- rowSums(edgeR::cpm(blood_y)>blood_cutoff) >= 3
blood_y <- blood_y[blood_keep, ,keep.lib.sizes=FALSE]
summary(blood_keep)
```

```
##    Mode   FALSE    TRUE
## logical   29452   33251
```

We remove 29452 rows (genes), almost 50% of the entire dataset.

6.4 MDS plot

```
par(mar=c(6,6,6,6))
plotMDS.DGEList(blood_y,col=as.double(sort(unique(blood_condition))))
par(xpd=T)
legend(par("usr")[2]*-1.4,par("usr")[4]*1,sort(unique(blood_condition)),
       pch=c(16),col=as.double(sort(unique(blood_condition))))
```



There does not seem to be a clear separation of log fold changes between healthy donors and COVID-19 patients.

It does seem, however, that COVID-19 patients' samples tend to be grouped together in the bottom of the MDS plot. Of course, due to the randomness of the healthy donors, we cannot say for sure that this is due to any effect.

7 Differential Expression Analysis

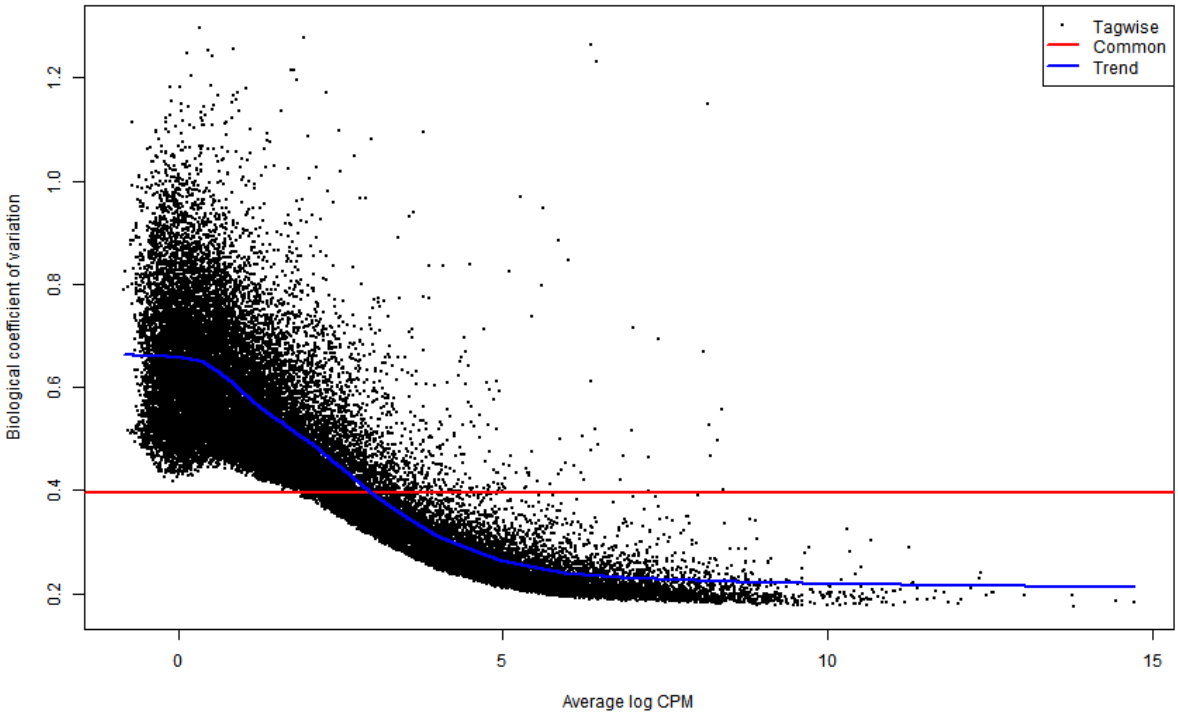
7.1 Make design matrix

```
blood_design <- model.matrix(~blood_condition)
rownames(blood_design) <- colnames(blood_y)
blood_design
```

```
##             (Intercept) blood_conditionHealthy
## SRR12926701           1                      1
## SRR12926702           1                      1
## SRR12926703           1                      1
## SRR12926705           1                      0
## SRR12926707           1                      0
## SRR12926708           1                      0
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$blood_condition
## [1] "contr.treatment"
```

7.2 Calculate dispersion

```
blood_y <- estimateDisp(blood_y,blood_design)
plotBCV(blood_y)
```



The dispersion contains high variation (around 1.0 - above 0.4), but this applies to low log CPM values below 4. Therefore, the biological variation in the data is acceptable.

7.3 Quasi-likelihood test

```
# Make contrast for quasi-likelihood test
blood_contrast <- makeContrast("-blood_conditionHealthy=0",
                               parameterNames = colnames(blood_design))
blood_fit <- glmQLFit(blood_y,blood_design)
blood_qlf <- glmQLFTest(blood_fit,contrast=blood_contrast)
blood_toptable <- topTags(blood_qlf,n=nrow(blood_qlf$table))$table
head(blood_toptable)
```

```
##                      logFC    logCPM        F       PValue          FDR
## ENSG00000133574 -2.852291  7.561370 213.0772 1.789466e-09 3.286572e-05
## ENSG00000183604 -3.192360  5.961228 209.6729 1.976826e-09 3.286572e-05
## ENSG00000075292 -2.672108  7.242392 171.6468 6.759532e-09 5.698432e-05
## ENSG00000245532 -2.569576 11.380310 167.2637 7.915031e-09 5.698432e-05
## ENSG00000137193  3.002630  6.863198 165.0990 8.568813e-09 5.698432e-05
## ENSG00000085224 -2.454092  7.603836 159.7308 1.047727e-08 5.806329e-05
```

```
sum(blood_toptable$FDR < 0.05)
```

```
## [1] 3916
```

There are 3916 significant genes found from the differential expression analysis.

7.4 Addition of gene symbols

7.4.1 Get gene symbols

```
data_sorted <- data[sort(data$ensembl_gene_id,index.return=T)$ix,]
data_sorted <- data_sorted[!duplicated(data_sorted$ensembl_gene_id),]
```

7.4.2 Add gene symbols to toptable

```
blood_toptable <- cbind(rownames(blood_toptable),blood_toptable)
colnames(blood_toptable)[1] <- "Ensembl_gene_id"
blood_toptable_sorted <- blood_toptable[sort(blood_toptable$Ensembl_gene_id,
                                             index.return=T)$ix,]
blood_data_sorted <- data_sorted[data_sorted$ensembl_gene_id %in%
                                 blood_toptable_sorted$Ensembl_gene_id,]
dim(blood_toptable)
```

```
## [1] 33251     6
```

```
dim(blood_data_sorted)
```

```
## [1] 33251     3
```

```
blood_toptable_sorted$Gene_symbol <- blood_data_sorted$external_gene_name
head(blood_toptable_sorted)
```

```
##                 Ensembl_gene_id       logFC    logCPM          F       PValue
## ENSG00000000003 ENSG00000000003  1.17168603 0.5600489  1.4927946 2.433520e-01
## ENSG00000000419 ENSG00000000419 -0.09298168 5.0198453  0.1566806 6.986137e-01
## ENSG00000000457 ENSG00000000457 -2.84043265 3.9011993 78.4806814 6.886433e-07
## ENSG00000000460 ENSG00000000460 -1.44138409 3.0522218  7.4605865 1.706036e-02
## ENSG00000000938 ENSG00000000938 -0.69454694 9.1606603 15.5539974 1.662918e-03
## ENSG00000000971 ENSG00000000971  1.01527165 3.2330418  6.7676473 2.185975e-02
##                          FDR Gene_symbol
## ENSG00000000003 0.4596771961      TSPAN6
## ENSG00000000419 0.8303998526        DPM1
## ENSG00000000457 0.0002187904       SCYL3
## ENSG00000000460 0.0955117667    C1orf112
## ENSG00000000938 0.0220030643         FGR
## ENSG00000000971 0.1109530992         CFH
```

7.4.3 Resort data

```
blood_toptable <- blood_toptable_sorted[sort(blood_toptable_sorted$PValue,
                                             index.return=T)$ix,]
```

7.5 Explore and save results

```
head(blood_toptable[,c(1,7,2,5,6)],10)
```

```
##                 Ensembl_gene_id Gene_symbol     logFC       PValue          FDR
## ENSG00000133574 ENSG00000133574      GIMAP4 -2.852291 1.789466e-09 3.286572e-05
## ENSG00000183604 ENSG00000183604      SMG1P5 -3.192360 1.976826e-09 3.286572e-05
## ENSG00000075292 ENSG00000075292      ZNF638 -2.672108 6.759532e-09 5.698432e-05
## ENSG00000245532 ENSG00000245532       NEAT1 -2.569576 7.915031e-09 5.698432e-05
## ENSG00000137193 ENSG00000137193        PIM1  3.002630 8.568813e-09 5.698432e-05
## ENSG00000085224 ENSG00000085224        ATRX -2.454092 1.047727e-08 5.806329e-05
## ENSG00000177853 ENSG00000177853     ZNF518A -3.037333 1.359433e-08 6.100015e-05
## ENSG00000271425 ENSG00000271425      NBPF10 -2.309884 1.467629e-08 6.100015e-05
## ENSG00000271533 ENSG00000271533             -3.905026 1.865306e-08 6.574824e-05
## ENSG00000291060 ENSG00000291060             -2.308543 1.977331e-08 6.574824e-05
```

```
write.table(blood_toptable,file="result/blood_toptable.txt",col.names=T,
            row.names=F,sep="\t",quote=F)
blood_toptable_sign <- blood_toptable[blood_toptable$FDR<0.05,]
dim(blood_toptable_sign)
```

```
## [1] 3916    7
```

Around 10% of genes found in the data are significantly expressed.

```
head(blood_toptable_sign,20)
```

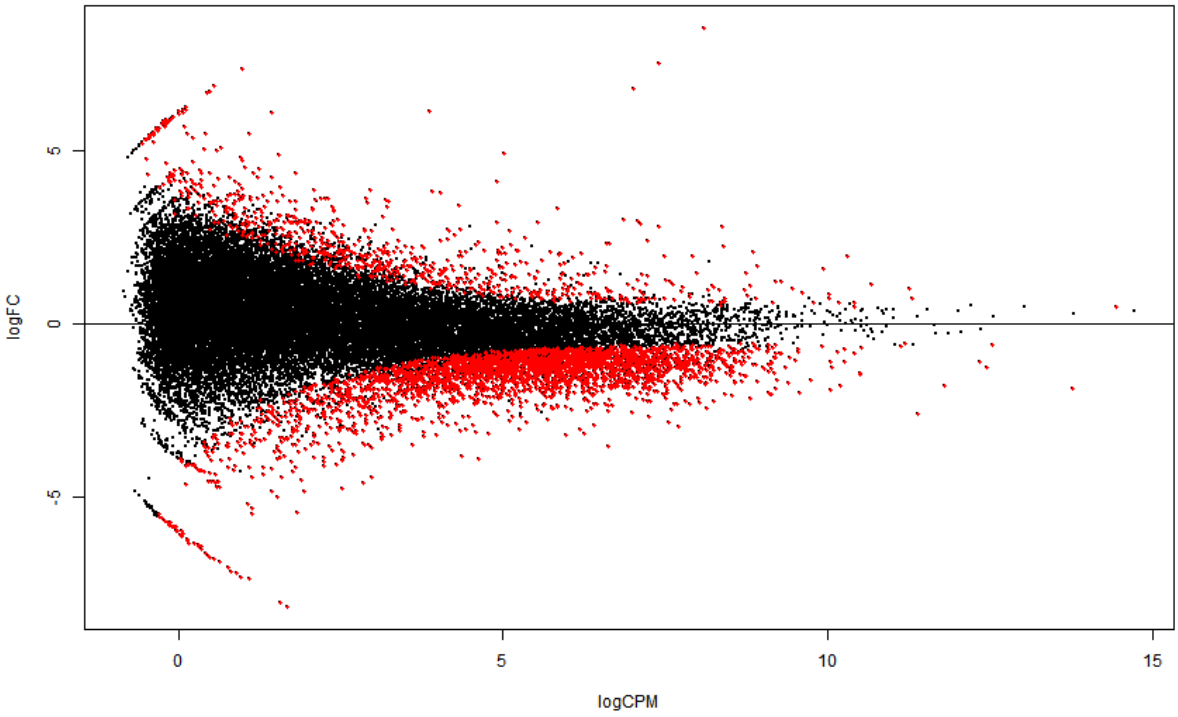
```
##                 Ensembl_gene_id     logFC    logCPM        F       PValue
## ENSG00000133574 ENSG00000133574 -2.852291  7.561370 213.0772 1.789466e-09
## ENSG00000183604 ENSG00000183604 -3.192360  5.961228 209.6729 1.976826e-09
## ENSG00000075292 ENSG00000075292 -2.672108  7.242392 171.6468 6.759532e-09
## ENSG00000245532 ENSG00000245532 -2.569576 11.380310 167.2637 7.915031e-09
## ENSG00000137193 ENSG00000137193  3.002630  6.863198 165.0990 8.568813e-09
## ENSG00000085224 ENSG00000085224 -2.454092  7.603836 159.7308 1.047727e-08
## ENSG00000177853 ENSG00000177853 -3.037333  6.176662 153.0187 1.359433e-08
## ENSG00000271425 ENSG00000271425 -2.309884  7.866640 151.0955 1.467629e-08
## ENSG00000271533 ENSG00000271533 -3.905026  4.622647 145.2174 1.865306e-08
## ENSG00000291060 ENSG00000291060 -2.308543  7.673726 143.8200 1.977331e-08
## ENSG00000129003 ENSG00000129003 -2.020899  9.132709 140.4345 2.282420e-08
## ENSG00000171115 ENSG00000171115 -3.535728  6.610462 134.5099 2.957017e-08
## ENSG00000119397 ENSG00000119397 -2.293864  7.630019 133.5115 3.092098e-08
## ENSG00000107290 ENSG00000107290 -2.057437  8.536159 132.5736 3.225496e-08
## ENSG00000127914 ENSG00000127914 -2.124212  7.999153 131.2756 3.421228e-08
## ENSG00000186088 ENSG00000186088 -2.291549  6.437892 131.1607 3.439199e-08
## ENSG00000145819 ENSG00000145819 -2.032230  8.425104 130.1022 3.610015e-08
## ENSG00000138640 ENSG00000138640 -2.395044  5.838538 126.3760 4.294262e-08
```

```
## ENSG00000198743 ENSG00000198743 -3.052136  5.256346 125.2207 4.535979e-08
## ENSG00000143669 ENSG00000143669 -1.982075  9.575178 124.2420 4.753097e-08
##                           FDR Gene_symbol
## ENSG00000133574 3.286572e-05      GIMAP4
## ENSG00000183604 3.286572e-05      SMG1P5
## ENSG00000075292 5.698432e-05      ZNF638
## ENSG00000245532 5.698432e-05       NEAT1
## ENSG00000137193 5.698432e-05        PIM1
## ENSG00000085224 5.806329e-05        ATRX
## ENSG00000177853 6.100015e-05     ZNF518A
## ENSG00000271425 6.100015e-05      NBPF10
## ENSG00000271533 6.574824e-05
## ENSG00000291060 6.574824e-05
## ENSG00000129003 6.899340e-05      VPS13C
## ENSG00000171115 7.060978e-05      GIMAP8
## ENSG00000119397 7.060978e-05       CNTRL
## ENSG00000107290 7.060978e-05        SETX
## ENSG00000127914 7.060978e-05       AKAP9
## ENSG00000186088 7.060978e-05        GSAP
## ENSG00000145819 7.060978e-05    ARHGAP26
## ENSG00000138640 7.566474e-05      FAM13A
## ENSG00000198743 7.566474e-05      SLC5A3
## ENSG00000143669 7.566474e-05        LYST
```

7.6 Plots

7.6.1 MA plot

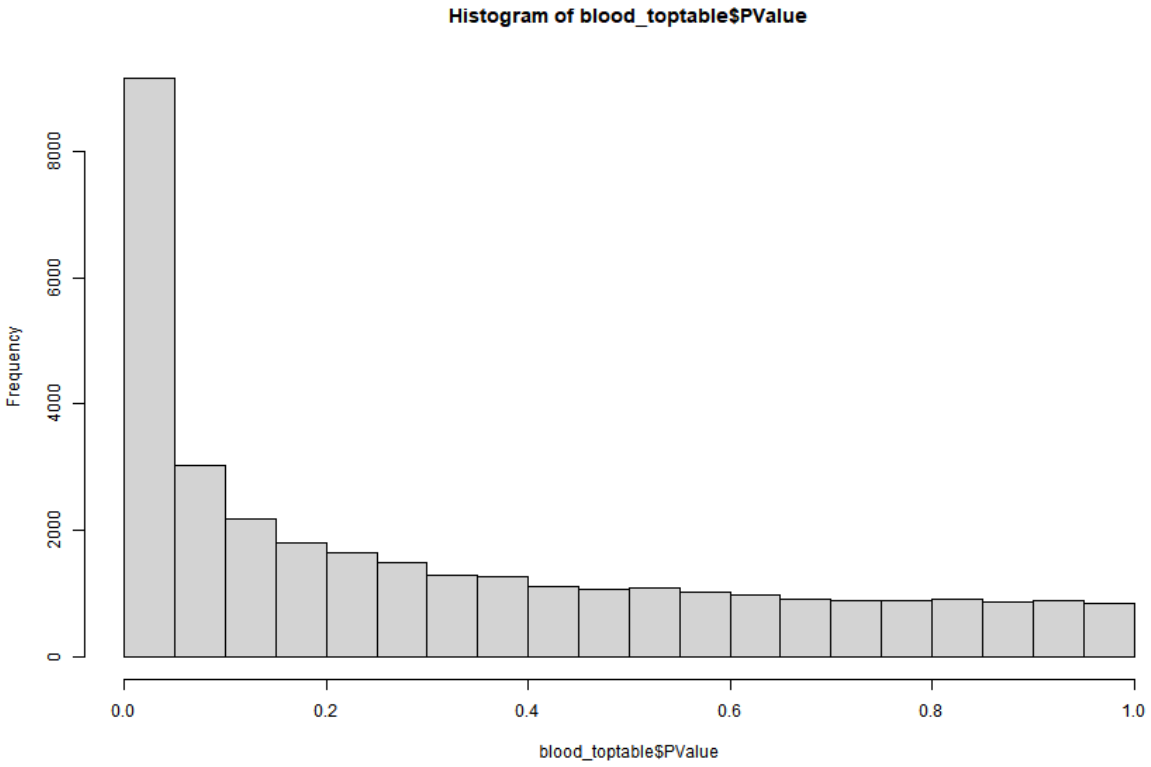
```
with(blood_toptable,plot(logCPM,logFC,pch=16,cex=0.2))
# MAplot: all data points
with(blood_toptable,points(logCPM[FDR<0.05],logFC[FDR<0.05],pch=16,col="red",cex=0.6))
# MA-plot: significant loci
abline(0,0)
```



There seems to be slightly more downregulated genes that is significant between healtyh donors and COVID-19 patients than upregulated genes. It can also be seen that there are a lot of significant genes.

7.6.2 P-value distribution

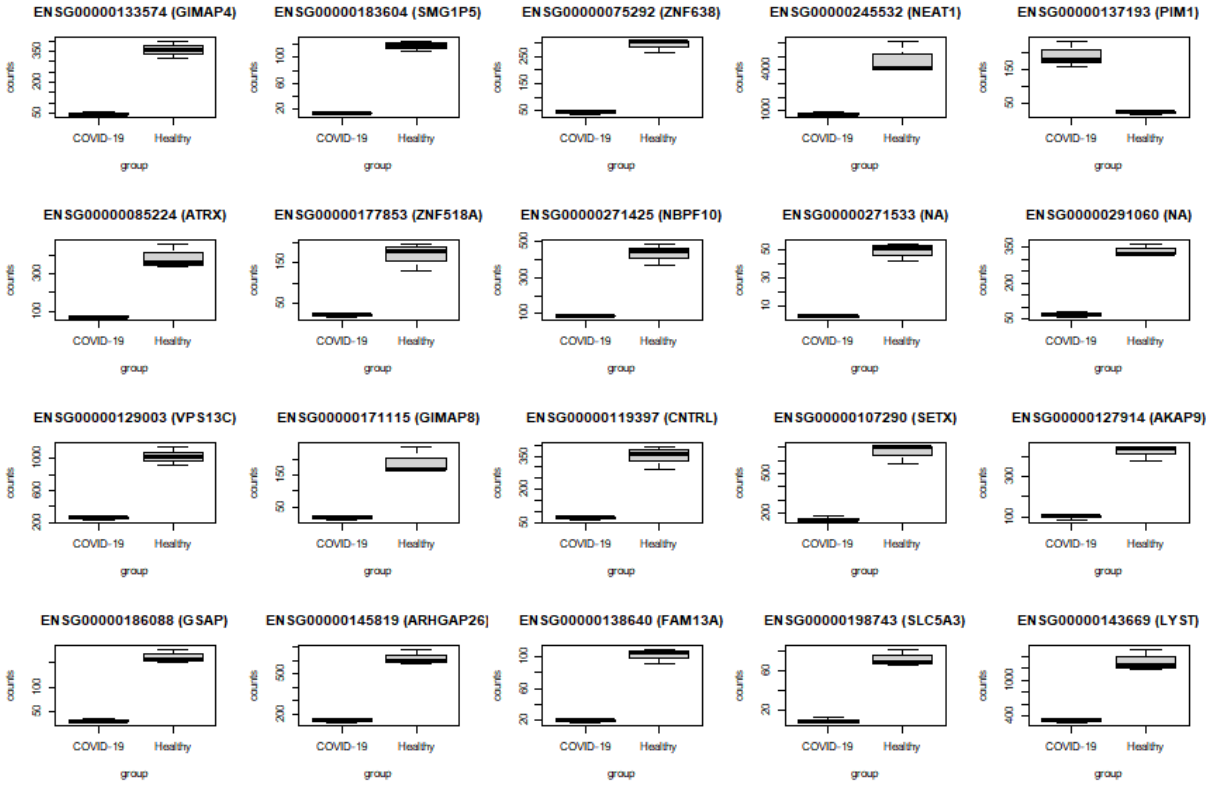
```
hist(blood_toptable$PValue)
```



The p-value distribution is not uniformly distributed.

7.6.3 Boxplots of top 20 loci

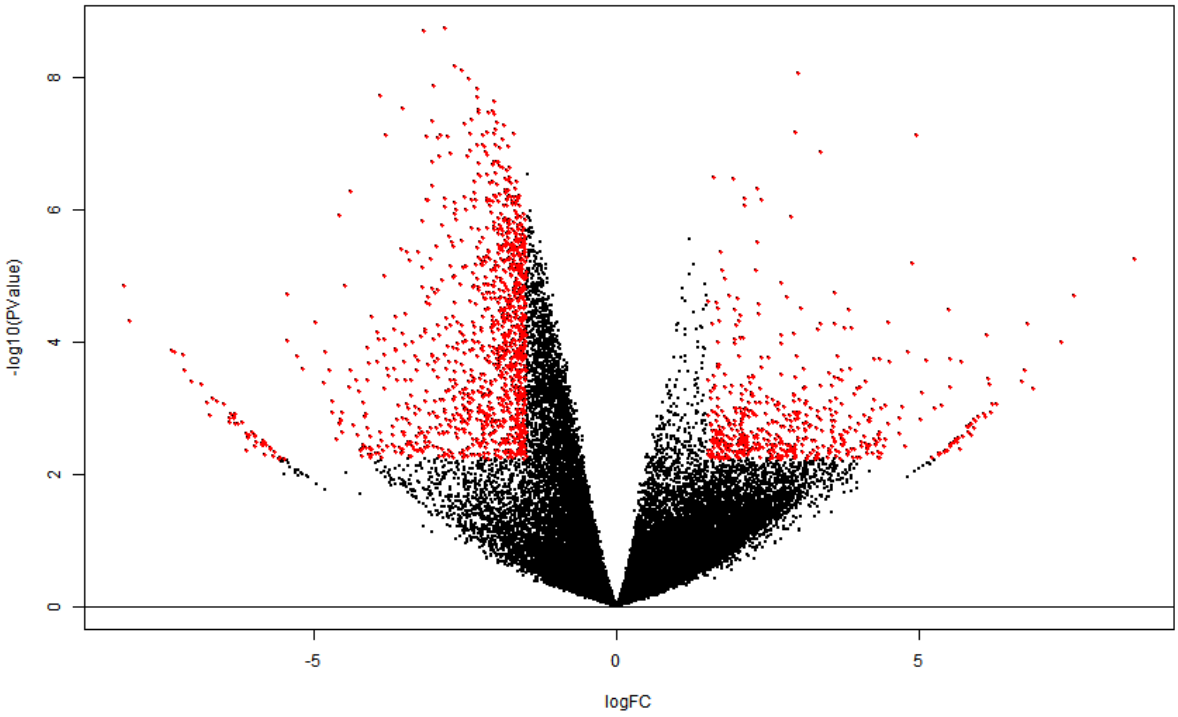
```
par(mfrow=c(4,5))
blood_counts_k <- blood_txi$counts[blood_keep,]
for (i in 1:20){
  blood_counts_part <- as.numeric(edgeR::cpm(blood_y)[rownames(blood_counts_k)==
                                                      rownames(blood_toptable)[i],])
  blood_boxplot <- data.frame(counts=blood_counts_part,group=blood_condition)
  if (blood_toptable$Gene_symbol[i]!=""){
    boxplot(counts~group,blood_boxplot,main=paste(rownames(blood_toptable)[i],
                                                  " (", blood_toptable$Gene_symbol[i],
                                                  ")",sep=""))
  } else {
    boxplot(counts~group,blood_boxplot,main=paste(rownames(blood_toptable)[i],
                                                  " (NA)", sep=""))
  }
}
```



The boxplots show that the the differentially expressed genes are indeed differentially expressed. In the first plot of `GIMPA4`, for example, there are no overlaps of the two box plots, and the two have around 300 count difference, with the gene underexpressed in COVID-19 patients.

7.6.4 Volcano plot

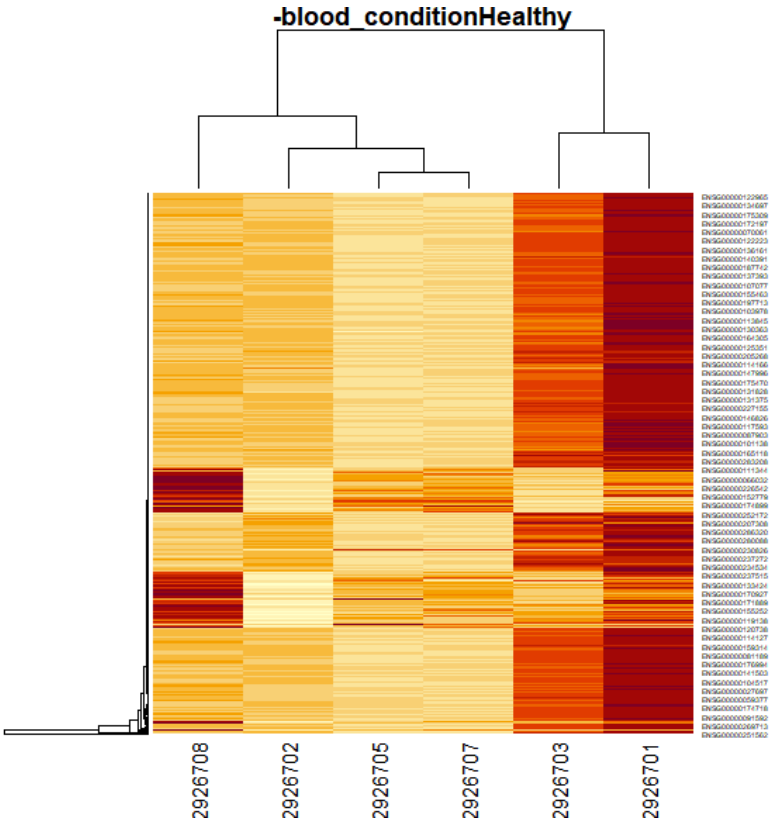
```
with(blood_toptable,plot(logFC,-log10(PValue),pch=16,cex=0.2))
with(blood_toptable,points(logFC[FDR<0.05 & abs(logFC)>1.5],
                           -log10(PValue)[FDR<0.05 & abs(logFC)>1.5],pch=16,
                           col="red",cex=0.6))
abline(0,0)
```



The volcano plot shows that there is slightly more underexpression than overexpression for differentially expressed genes.

7.6.5 Heatmap

```
blood_sigID <- blood_toptable_sign %>% rownames
heatmap(blood_y$counts[blood_sigID,], main = colnames(blood_contrast),
        cex.main=.2)
```



8 Gene set analysis

From the differential expression analysis, we found a lot of differentially expressed genes. Therefore, we do further analysis with gene set analysis. Specifically, we make use of overrepresentation analysis (ORA).

8.1 Annotation

For annotation, the `data` from Section 1, human annotation, is used. Duplicated ensembl gene IDs are removed and the data is saved in `.rda` file.

```
if(!file.exists("RefGenes.Rda")){
  refgenes <- data[!duplicated(data$ensembl_gene_id),]
  refgenes <- refgenes[c("ensembl_gene_id")]
  write.table(refgenes,file="RefGenes.txt", col.names=c("Ensembl_gene_id"),
              row.names=F, quote=F, sep='\t')
} else{
  load(refgenes, file="RefGenes.txt")
}
```

8.2 Gene set analysis using custom gene sets

8.2.1 Make `gmt` file for `WebGestaltR`

The file format `gmt` should contain the gene sets.

```
if (!file.exists("geneset_full.gmt")){
  geneset <- read.csv("geneset_full.csv", header=F)
  write.table(geneset, file="geneset_full.gmt",col.names=T,row.names=F,sep="\t",
              quote=F,na="")
}
```

8.2.2 Run WebGestaltR using custom gene sets

```
blood_enrichResult_full <- WebGestaltR(enrichMethod="ORA",
                            organism="hsapiens",
                            enrichDatabase="others",
                            enrichDatabaseFile="geneset_full.gmt",
                            enrichDatabaseType="genesymbol",
                            interestGeneFile="result/blood_toptable.txt",
                            interestGeneType="ensembl_gene_id",
                            referenceGeneFile="RefGenes.txt",
                            referenceGeneType="ensembl_gene_id",
                            sigMethod="fdr",
                            fdrThr=0.05,
                            outputDirectory='result',
                            projectName = 'ORA_blood',
                            is.ouput=TRUE)
```

8.3 Gene set analysis using KEGG pathway

```
blood_enrichResult_kegg <- WebGestaltR(enrichMethod="ORA",
                            organism="hsapiens",
                            enrichDatabase="pathway_KEGG",
                            interestGeneFile="result/blood_toptable.txt",
                            interestGeneType="ensembl_gene_id",
                            referenceGeneFile="RefGenes.txt",
                            referenceGeneType="ensembl_gene_id",
                            sigMethod="fdr",
                            fdrThr=0.01,
                            outputDirectory='result',
                            projectName = 'ORA_blood_kegg_0.01',
                            is.ouput=TRUE)
```

C.3 Results

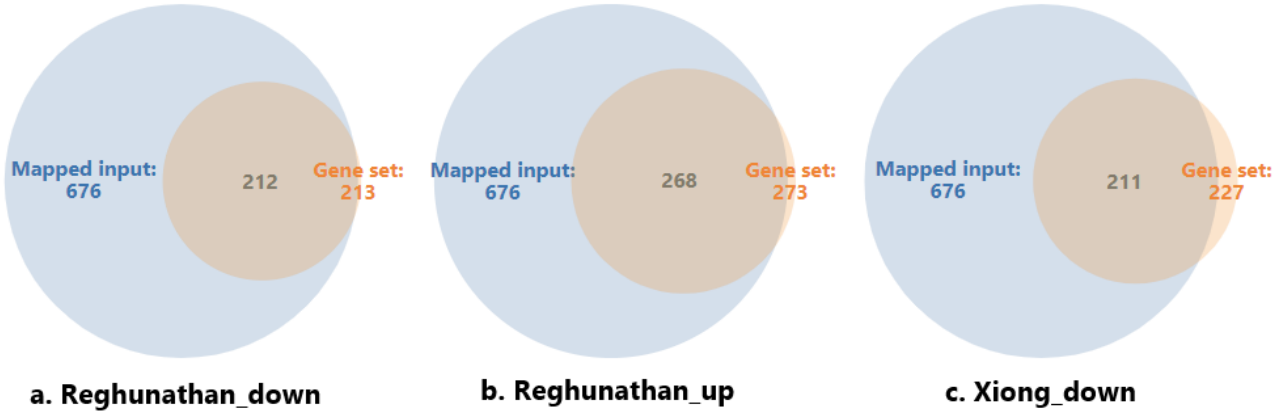


Figure S10: Overlaps between the mapped genes from monocyte RNA-seq data and the (a) downregulated genes from Reghunathan et al., (b) upregulated genes from Reghunathan et al., and (c) downregulated genes from Xiong et al.

Table S2: *Gene set analysis result of the monocyte RNA-seq data using KEGG pathway with a threshold at FDR below 0.01.*

Gene Set	Description	Size	Expect	Ratio	*P-value*	*FDR*
hsa04144	Endocytosis	244	218.42	1.1079	5.3335e-10	1.6800e-7
hsa04360	Axon guidance	175	156.65	1.1171	3.0007e-9	4.7262e-7
hsa04210	Apoptosis	136	121.74	1.1171	2.4851e-7	0.000026093
hsa05166	Human T-cell leukemia virus 1 infection	255	228.27	1.0908	4.5718e-7	0.000036003
hsa04714	Thermogenesis	229	204.99	1.0927	0.0000010025	0.000056552
hsa04142	Lysosome	123	110.11	1.1171	0.0000010772	0.000056552
hsa04071	Sphingolipid signaling pathway	118	105.63	1.1171	0.0000018921	0.000085145
hsa04934	Cushing syndrome	154	137.86	1.1026	0.0000061195	0.00023569
hsa04068	FoxO signaling pathway	131	117.27	1.1086	0.0000072746	0.00023569
hsa04921	Oxytocin signaling pathway	152	136.07	1.1024	0.0000074821	0.00023569
hsa03010	Ribosome	151	135.17	1.1023	0.0000082721	0.00023688
hsa04141	Protein processing in endoplasmic reticulum	165	147.70	1.0968	0.000013841	0.00034492
hsa04110	Cell cycle	124	111.00	1.1081	0.000015201	0.00034492
hsa04510	Focal adhesion	199	178.14	1.0890	0.000015330	0.00034492
hsa04611	Platelet activation	123	110.11	1.1080	0.000016884	0.00035455
hsa01522	Endocrine resistance	96	85.936	1.1171	0.000022457	0.00044212
hsa05016	Huntington disease	192	171.87	1.0880	0.000028504	0.00052816
hsa04120	Ubiquitin mediated proteolysis	137	122.64	1.1008	0.000033373	0.00057222
hsa04919	Thyroid hormone signaling pathway	116	103.84	1.1075	0.000035134	0.00057222
hsa04010	MAPK signaling pathway	295	264.08	1.0717	0.000036331	0.00057222
hsa03040	Spliceosome	134	119.95	1.1004	0.000044877	0.00061462
hsa00240	Pyrimidine metabolism	101	90.412	1.1060	0.00016661	0.0018966
hsa04070	Phosphatidylinositol signaling system	99	88.622	1.1058	0.00020471	0.0018966
hsa05231	Choline metabolism in cancer	99	88.622	1.1058	0.00020471	0.0018966
hsa05220	Chronic myeloid leukemia	76	68.033	1.1171	0.00021141	0.0019027
hsa04211	Longevity regulating pathway	88	78.775	1.1044	0.00063054	0.0038945
hsa04012	ErbB signaling pathway	85	76.089	1.1040	0.00085483	0.0049865
hsa04512	ECM-receptor interaction	82	73.404	1.1035	0.0011575	0.0062867
hsa05012	Parkinson disease	142	127.11	1.0778	0.0019034	0.0084447
hsa04668	TNF signaling pathway	110	98.469	1.0866	0.0021496	0.0091501
hsa05130	Pathogenic Escherichia coli infection	55	49.234	1.1171	0.0022109	0.0091634

D ChIP-seq Data Analysis

This section gives details on the data analysis of ChIP-seq data and contains the analysis results.

D.1 Quality Control

FastQC was used for quality control and the FastQC results were combined via MultiQC. The following shows some of the relevant figures from quality control.

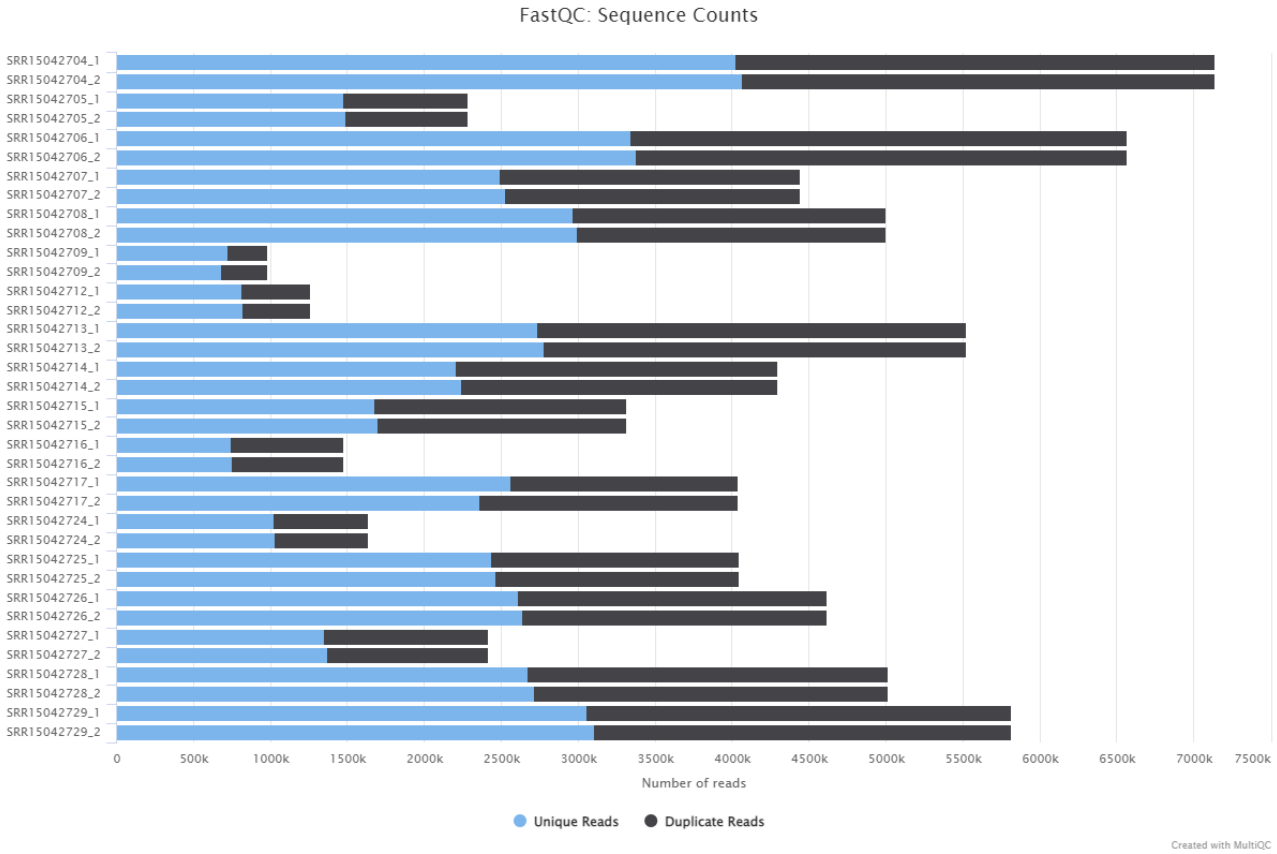


Figure S11: Sequence counts for each sample of ChIP RNA-seq data. Duplicate read counts are an estimate only. The extensions after the sample names indicate the forward read (*1*) and reverse read (*2*).

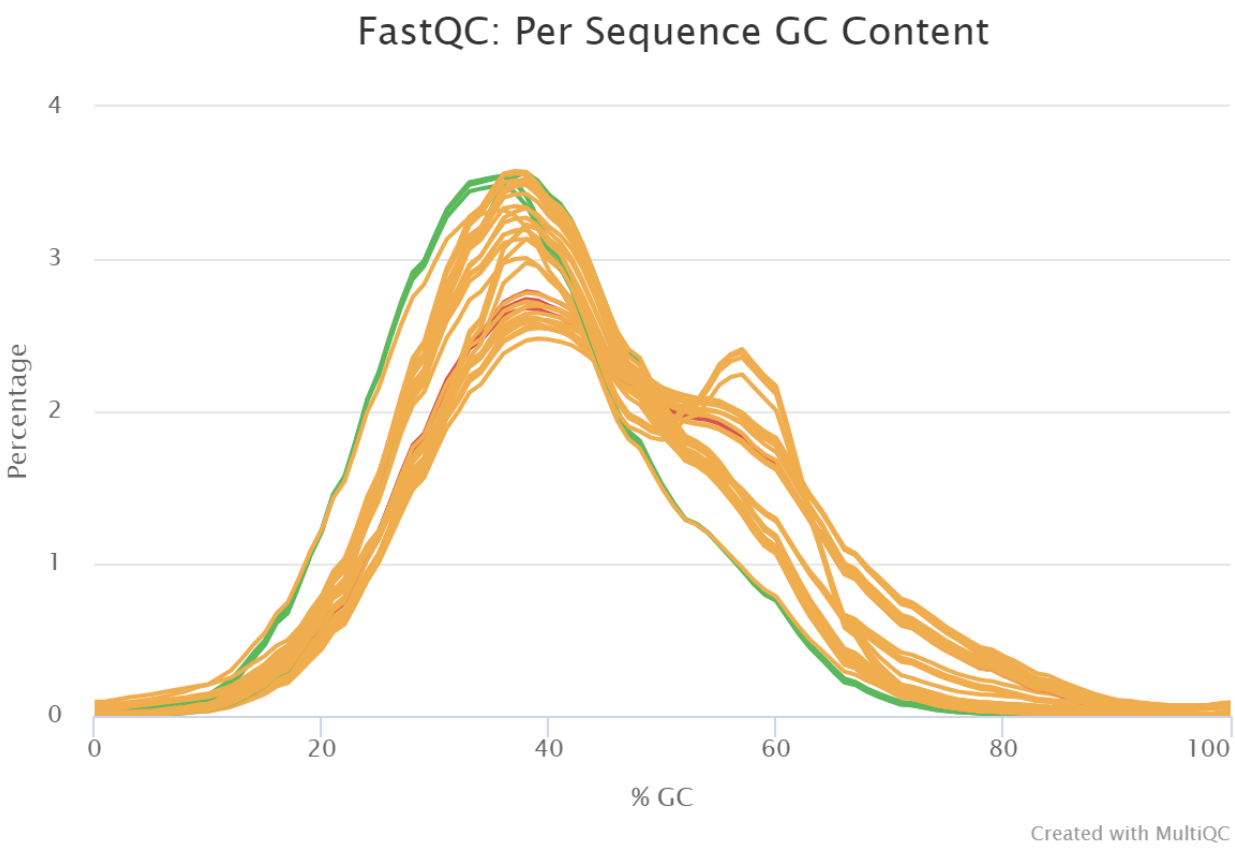


Figure S12: Per Sequence GC Content of ChIP RNA-seq data. The average GC content of reads.

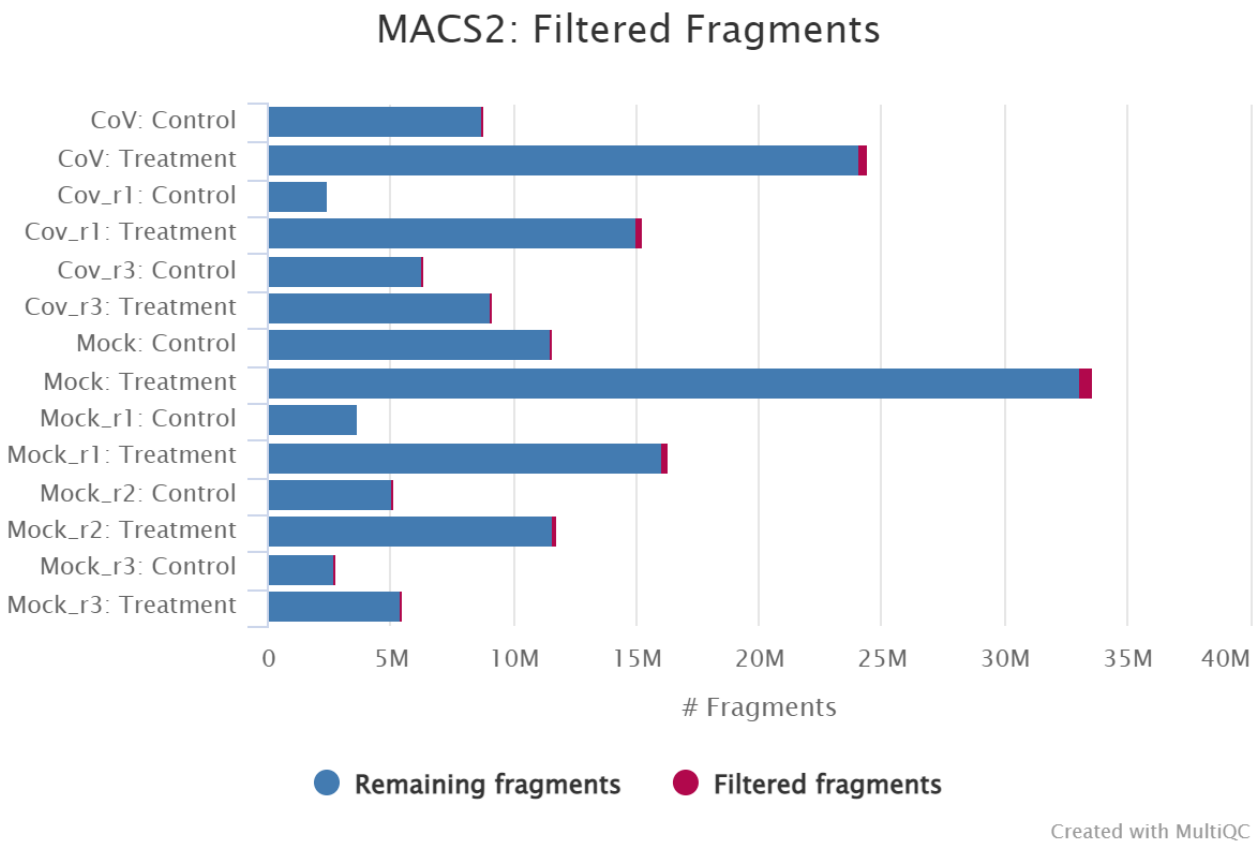


Figure S13: MACS2 filtered fragments of ChIP RNA-seq data. *Mock* refers to mock-treated samples and *CoV* refers to SARS-CoV-2 treated samples. *Control* refers to the input control for ChIP-seq and *Treatment* refers to the samples. The samples with *r1*, *r2*, *r3* in their names refers to the peaks of each replicates. Samples without any replicate numbers are merged peaks.

D.2 R Markdown of Data Analysis

The following is the R markdown file of the statistical analysis of ChIP-seq Data.

1 Load libraries

```
suppressPackageStartupMessages({
  library("dplyr")
  library("GEOquery")
  library("GenomicRanges")
  library("TxDb.Hsapiens.UCSC.hg38.knownGene")
  library("org.Hs.eg.db")
  library("DiffBind")
})
```

2 Load data

2.1 Peak data of mock treatment

```
## Load and annotate data
mock_beddata<-read.table("./Data/chipseq/Mock_peaks.broadPeak",header=F)
colnames(mock_beddata)<-c("seqnames","start","end","id","score","strand",
                          "enrichment","log10p","log10q")

## Adjust strand data
mock_beddata$seqnames <- paste0("chr", mock_beddata$seqnames)
mock_beddata$strand<-as.factor("*")
head(mock_beddata)
```

```
##   seqnames  start    end                id score strand enrichment   log10p
## 1     chr1  28018  29957 peaks/Mock_peak_1   309      *   10.38390 34.48590
## 2     chr1 198427 200402 peaks/Mock_peak_2   407      *    9.06014 44.45460
## 3     chr1 777822 780437 peaks/Mock_peak_3   251      *    8.71534 28.63860
## 4     chr1 826003 828554 peaks/Mock_peak_4   207      *    8.95931 24.10090
## 5     chr1 872785 873020 peaks/Mock_peak_5    10      *    3.33777  3.96879
## 6     chr1 903722 906756 peaks/Mock_peak_6    85      *    6.09756 11.67230
##     log10q
## 1 30.95430
## 2 40.74580
## 3 25.17050
## 4 20.73850
## 5  1.09895
## 6  8.52752
```

```
dim(mock_beddata)
```

```
## [1] 35596     9
```

```
write.table(mock_beddata,file="data/chipseq/Mock.bed",col.names = T,
            row.names = F,quote = F,sep="\t")
```

2.2 Peak data of Sars-Cov-2 infection

```
## Load and annotate data
covid_beddata<-read.table("./Data/chipseq/Cov_peaks.broadPeak",header=F)
colnames(covid_beddata)<-c("seqnames","start","end","id","score","strand",
                           "enrichment","log10p","log10q")

## Adjust strand data
covid_beddata$seqnames <- paste0("chr", covid_beddata$seqnames)
covid_beddata$strand<-as.factor("*")
head(covid_beddata)
```

```
##   seqnames  start    end               id score strand enrichment   log10p
## 1     chr1  28116  29904 peaks/CoV_peak_1   261      *   12.39690 29.99080
## 2     chr1 198560 200459 peaks/CoV_peak_2   314      *   13.06120 35.38790
## 3     chr1 778019 780144 peaks/CoV_peak_3   174      *    9.45836 21.02750
## 4     chr1 826023 828550 peaks/CoV_peak_4   134      *    8.13709 16.90210
## 5     chr1 904041 905612 peaks/CoV_peak_5    49      *    5.33891  8.14113
## 6     chr1 923604 926387 peaks/CoV_peak_6    55      *    5.55629  8.73243
##     log10q
## 1 26.18800
## 2 31.45930
## 3 17.46790
## 4 13.45810
## 5  4.95784
## 6  5.53326
```

```
dim(covid_beddata)
```

```
## [1] 25028     9
```

```
write.table(covid_beddata,file="data/chipseq/Cov.bed",col.names = T,
            row.names = F,quote = F,sep="\t")
```

2.3 Annotation

Our interest lays in the differential prevalence of the H3Kac histone modification (per DNA region) between treatments.

```
chip_geo <- getGEO("GSE205369")
chip_geo <- phenoData(chip_geo[["GSE205369_series_matrix.txt.gz"]])@data[c("title", "geo_accession")]
chip_annotation <- read.table("chip_annotation.txt", header=T, sep=",")
chip_annotation <- chip_annotation %>% dplyr::rename(geo_accession = `Library.Name`)
chip_annotation <- dplyr::filter(chip_annotation, Run %in% c("SRR19522212",
"SRR19522213","SRR19522214","SRR19522215","SRR19522216","SRR19522223",
"SRR19522224","SRR19522225","SRR19522226","SRR19522227"))
chip_geo <- chip_geo[chip_geo$geo_accession %in% chip_annotation$geo_accession,]
chip_geo$title <- chip_geo$title %>% strsplit(" ") %>% vapply("[","",9)
chip_geo <- chip_geo %>% dplyr::rename(replicate = `title`)
chip_annotation <- merge(chip_annotation, chip_geo, "geo_accession")
print(chip_annotation[c("Run", "chip_target", "Treatment", "Organism",
                    "Cell_type", "AvgSpotLen", "Instrument", "LibraryLayout")])
```

```
##            Run                          chip_target      Treatment     Organism
## 1  SRR19522227                        Input control Mock infection Homo sapiens
## 2  SRR19522226                        Input control Mock infection Homo sapiens
## 3  SRR19522225                        Input control Mock infection Homo sapiens
## 4  SRR19522224                        Input control CoV2 infection Homo sapiens
## 5  SRR19522223                        Input control CoV2 infection Homo sapiens
## 6  SRR19522216 H3K9ac IP (Active Motif, cat# 39137) Mock infection Homo sapiens
## 7  SRR19522215 H3K9ac IP (Active Motif, cat# 39137) Mock infection Homo sapiens
## 8  SRR19522214 H3K9ac IP (Active Motif, cat# 39137) Mock infection Homo sapiens
## 9  SRR19522213 H3K9ac IP (Active Motif, cat# 39137) CoV2 infection Homo sapiens
## 10 SRR19522212 H3K9ac IP (Active Motif, cat# 39137) CoV2 infection Homo sapiens
##    Cell_type AvgSpotLen  Instrument LibraryLayout
## 1       A549         83 NextSeq 550        PAIRED
## 2       A549         82 NextSeq 550        PAIRED
## 3       A549         82 NextSeq 550        PAIRED
## 4       A549         82 NextSeq 550        PAIRED
## 5       A549         82 NextSeq 550        PAIRED
## 6       A549         83 NextSeq 550        PAIRED
## 7       A549         83 NextSeq 550        PAIRED
## 8       A549         83 NextSeq 550        PAIRED
## 9       A549         83 NextSeq 550        PAIRED
## 10      A549         82 NextSeq 550        PAIRED
```

3 Analysis

Here, we try to find genes of interest.

```
# Create GRanges objects
mock_ChIPGR <- with(mock_beddata,
                    GRanges(seqnames, IRanges(start+1, end), strand, score, id,
                            enrichment, log10p, log10q))
covid_ChIPGR <- with(covid_beddata,
                     GRanges(seqnames, IRanges(start+1, end), strand, score, id,
                             enrichment, log10p, log10q))
```

```
# Get gene data
humangenes <- genes(TxDb.Hsapiens.UCSC.hg38.knownGene)
```

```
# Get overlaps
mock_ChIPgenes <- subsetByOverlaps(humangenes,mock_ChIPGR,ignore.strand=T)
head(mock_ChIPgenes)
```

```
## GRanges object with 6 ranges and 1 metadata column:
##             seqnames              ranges strand |     gene_id
##                <Rle>           <IRanges>  <Rle> | <character>
##           1    chr19   58345178-58362751      - |           1
##         100    chr20   44619522-44652233      - |         100
##        1000    chr18   27932879-28177946      - |        1000
##   100009676     chr3 101676475-101679217      + |   100009676
##       10001    chr14   70581257-70641204      - |       10001
##       10002    chr15   71792638-71818259      + |       10002
```

```
##   -------
##   seqinfo: 640 sequences (1 circular) from hg38 genome
```

```
covid_ChIPgenes <- subsetByOverlaps(humangenes,covid_ChIPGR,ignore.strand=T)
head(covid_ChIPgenes)
```

```
## GRanges object with 6 ranges and 1 metadata column:
##             seqnames              ranges strand |     gene_id
##                <Rle>           <IRanges>  <Rle> | <character>
##           1    chr19   58345178-58362751      - |           1
##         100    chr20   44619522-44652233      - |         100
##        1000    chr18   27932879-28177946      - |        1000
##   100009676     chr3 101676475-101679217      + |   100009676
##       10001    chr14   70581257-70641204      - |       10001
##       10002    chr15   71792638-71818259      + |       10002
##   -------
##   seqinfo: 640 sequences (1 circular) from hg38 genome
```

```
# Gene annotation
columns(org.Hs.eg.db)
```

```
##  [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS"
##  [6] "ENTREZID"     "ENZYME"       "EVIDENCE"     "EVIDENCEALL"  "GENENAME"
## [11] "GENETYPE"     "GO"           "GOALL"        "IPI"          "MAP"
## [16] "OMIM"         "ONTOLOGY"     "ONTOLOGYALL"  "PATH"         "PFAM"
## [21] "PMID"         "PROSITE"      "REFSEQ"       "SYMBOL"       "UCSCKG"
## [26] "UNIPROT"
```

```
mock_ChIPgenes <- select(org.Hs.eg.db, mock_ChIPgenes$gene_id,
                         c("SYMBOL", "GENENAME"))
colnames(mock_ChIPgenes) <- c("Entrez_ID","Gene_Symbol","Gene_Name")
head(mock_ChIPgenes)
```

```
##   Entrez_ID Gene_Symbol                                    Gene_Name
## 1         1        A1BG                      alpha-1-B glycoprotein
## 2       100         ADA                         adenosine deaminase
## 3      1000        CDH2                                   cadherin 2
## 4 100009676  ZBTB11-AS1                      ZBTB11 antisense RNA 1
## 5     10001        MED6                   mediator complex subunit 6
## 6     10002       NR2E3 nuclear receptor subfamily 2 group E member 3
```

```
covid_ChIPgenes <- select(org.Hs.eg.db, covid_ChIPgenes$gene_id,
                          c("SYMBOL", "GENENAME"))
colnames(covid_ChIPgenes) <- c("Entrez_ID","Gene_Symbol","Gene_Name")
head(covid_ChIPgenes)
```

```
##   Entrez_ID Gene_Symbol                                    Gene_Name
## 1         1        A1BG                      alpha-1-B glycoprotein
## 2       100         ADA                         adenosine deaminase
## 3      1000        CDH2                                   cadherin 2
## 4 100009676  ZBTB11-AS1                      ZBTB11 antisense RNA 1
## 5     10001        MED6                   mediator complex subunit 6
## 6     10002       NR2E3 nuclear receptor subfamily 2 group E member 3
```

```
# Write results to tables
write.table(mock_ChIPgenes,file="result/mock_ChIPgenes.txt",col.names = T,
            row.names = F,quote = F,sep="\t")
write.table(covid_ChIPgenes,file="result/covid_ChIPgenes.txt",col.names = T,
            row.names = F,quote = F,sep="\t")
```

```
# Find genes that are different between mock and control
diff <- setdiff(covid_ChIPgenes$Gene_Symbol, mock_ChIPgenes$Gene_Symbol)
head(diff)
```

```
## [1] "MIR3677"      "MIR3648-1"    "LOC100506446" "LINC02274"    "MIR3680-2"
## [6] "MIR4999"
```

```
length(diff)
```

```
## [1] 128
```

4 Visualization

```
# Subset only ones with correct chromosome names
mock_subset <- mock_beddata[mock_beddata$seqnames %in%
                              paste0("chr", c(1:22, "X", "Y")),]
mock_subset$strand <- "."
covid_subset <- covid_beddata[covid_beddata$seqnames %in%
                                paste0("chr", c(1:22, "X", "Y")),]
covid_subset$strand <- "."
```

```
# Make track files
write('track type=broadPeak visibility=3 db=hg38 name="Mock" description="Mock treatment"',
      file = "result/Mock_track.broadPeak")
write.table(mock_subset, file = "result/Mock_track.broadPeak", append=T,
            sep = "\t", quote=F, row.names=F, col.names=F)
write('track type=broadPeak visibility=3 db=hg38 name="Covid" description="Covid treatment"',
      file = "result/Covid_track.broadPeak")
write.table(covid_subset, file = "result/Covid_track.broadPeak", append=T,
            sep = "\t", quote=F, row.names=F, col.names=F)
```

5 Differential Enrichment Analysis

5.1 Make `dba` object

In the sections above, peak data of all replicates merged were used. For the differential enrichment analysis, the dba function required sample bam files and their respective peak files. Therefore, we make use of peaks created for each replicate, and the datasets are not exactly the same as the one found above.

```
if(file.exists('data/chipseq/sample_sheet.csv')){
  sample_sheet <- read.table('data/chipseq/sample_sheet.csv', header=T, sep=',')
} else {
  samples <- chip_annotation[!chip_annotation$chip_target=='Input control',]
  control <- chip_annotation[chip_annotation$chip_target=='Input control',]
  treatment <- strsplit(samples$Treatment, ' ') %>% vapply('[', '', 1)
  sample_sheet <- data.frame(SampleID = paste(treatment, samples$replicate, sep='.'),
                             Factor = treatment,
                             Tissue = samples$Cell_type,
                             Condition = treatment,
                             Replicate = strsplit(samples$replicate, 'r') %>%
                                         vapply('[','',2),
                             bamReads = paste0('data/chipseq/', samples$Run,
                                               '_sored.bam'),
                             bamControl = paste0('data/chipseq/', control$Run,
                                                 '_sored.bam'),
                             ControlID = control$Run,
                             Peaks = c("data/chipseq/Mock_r1_peaks.xls",
                                       "data/chipseq/Mock_r2_peaks.xls",
                                       "data/chipseq/Mock_r3_peaks.xls",
                                       "data/chipseq/Cov_r1_peaks.xls",
                                       "data/chipseq/Cov_r3_peaks.xls"),
                             PeakCaller = unlist(lapply(1:5, function(x) "macs")),
                             PeakFormat = unlist(lapply(1:5, function(x) "macs")))
  write.table(sample_sheet, 'data/chipseq/sample_sheet.csv', col.names = T,
              row.names = F, quote = F, sep=',')
}
dbObj <- dba(sampleSheet = sample_sheet)

dbObj[["peaks"]][[1]]$Chr <- paste0("chr", dbObj[["peaks"]][[1]]$Chr)
dbObj[["peaks"]][[2]]$Chr <- paste0("chr", dbObj[["peaks"]][[2]]$Chr)
dbObj[["peaks"]][[3]]$Chr <- paste0("chr", dbObj[["peaks"]][[3]]$Chr)
dbObj[["peaks"]][[4]]$Chr <- paste0("chr", dbObj[["peaks"]][[4]]$Chr)
dbObj[["peaks"]][[5]]$Chr <- paste0("chr", dbObj[["peaks"]][[5]]$Chr)

dbObj
```

```
## 5 Samples, 20642 sites in matrix (29472 total):
##        ID Tissue Factor Condition Replicate Intervals
## 1 Mock.r1   A549   Mock      Mock         1     20009
## 2 Mock.r2   A549   Mock      Mock         2     25904
## 3 Mock.r3   A549   Mock      Mock         3     19515
## 4 CoV2.r1   A549   CoV2      CoV2         1     15098
## 5 CoV2.r3   A549   CoV2      CoV2         3     18066
```

5.2 Create affinity binding matrix

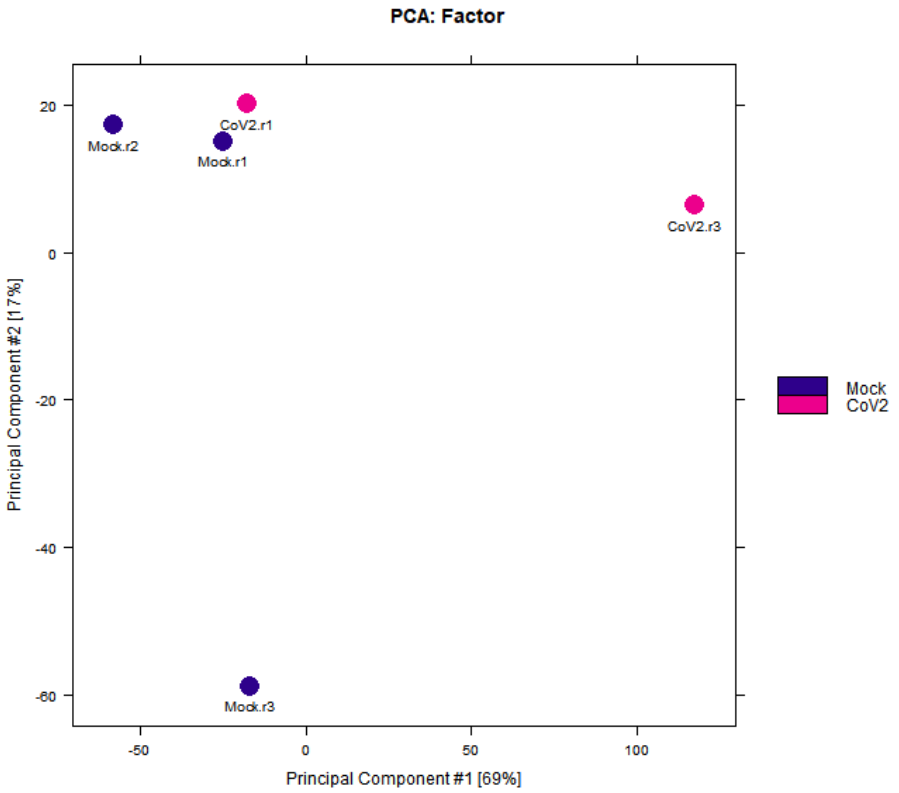
```
dbObj <- dba.count(dbObj, bUseSummarizeOverlaps=TRUE)
dbObj
```

```
## 5 Samples, 19124 sites in matrix:
```

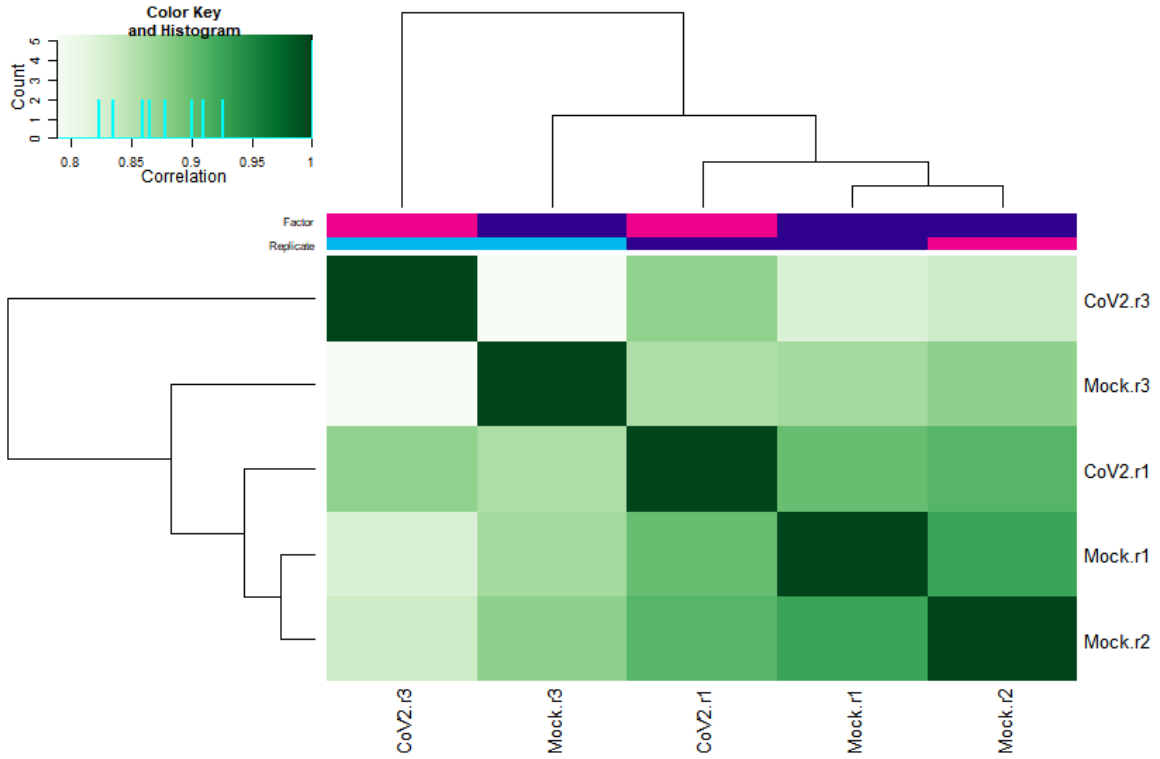
```
##        ID Tissue Factor Condition Replicate    Reads FRiP
## 1 Mock.r1   A549   Mock      Mock         1 15096904 0.09
## 2 Mock.r2   A549   Mock      Mock         2 10960652 0.12
## 3 Mock.r3   A549   Mock      Mock         3  5038242 0.09
## 4 CoV2.r1   A549   CoV2      CoV2         1 14098487 0.09
## 5 CoV2.r3   A549   CoV2      CoV2         3  8337580 0.06
```

5.3 PCA

```
dba.plotPCA(dbObj, attributes=DBA_FACTOR, label=DBA_ID)
```




```
plot(dbObj)
```



While there are correlations between Cov treated samples and mock treated samples, there seems to be more correlation between the replicates.

5.4 Make contrast

The replicates are used as block effect.

```
dbObj <- dba.contrast(dbObj, design=FALSE, categories=DBA_FACTOR,
                      block=DBA_REPLICATE, minMembers = 2)
```

5.5 Analyze

Greylist parameter was removed due to unknown reasons.

```
dbObj <- dba.analyze(dbObj, method=DBA_ALL_METHODS, bGreylist=F)
```

5.6 Result exploration

5.6.1 Contrast

```
dba.show(dbObj, bContrasts=T)
```

```
##   Group Samples Group2 Samples2 Block1 Blk1Samps Block2 Blk2Samps Block3
## 1  Mock       3   CoV2        2      1         2      2         1      3
##   Blk3Samps DB.edgeR DB.edgeR.block DB.DESeq2 DB.DESeq2.block
## 1         2      510            305         7               0
```

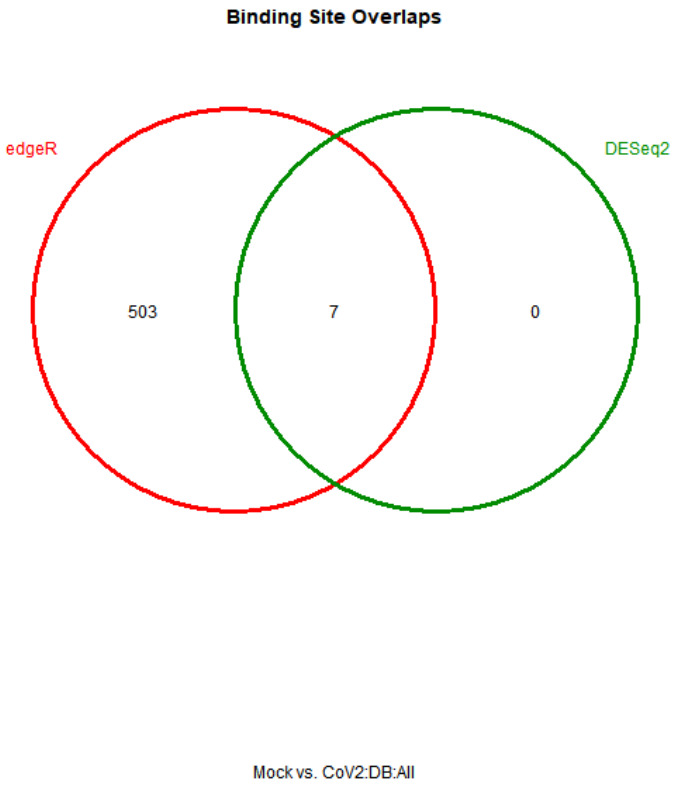
5.6.2 Plots

```
jpeg("images/chip_pca.jpg")
dba.plotPCA(dbObj, contrast=1, method=DBA_DESEQ2, attributes=DBA_FACTOR, label=DBA_ID)
dev.off()
```

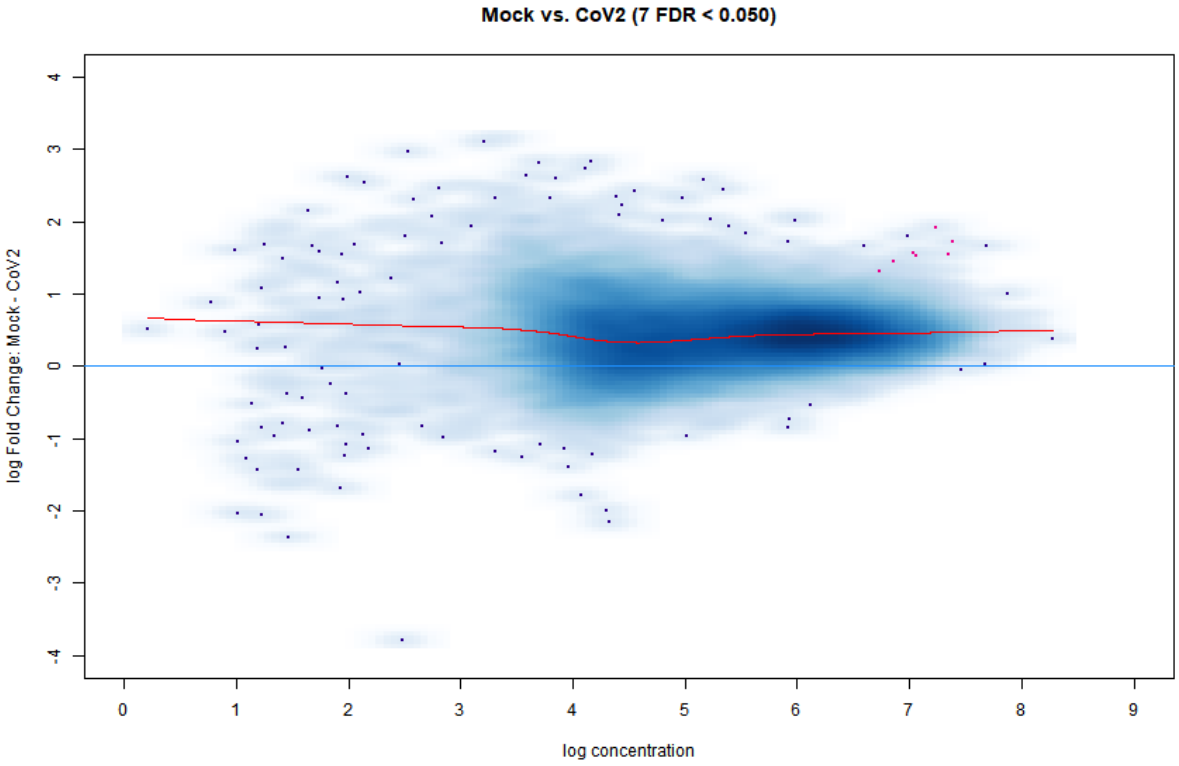
```
## png
##   2
```

The PCA shows grouping of the mock treatment and cov2 treatment samples.

```
dba.plotVenn(dbObj,contrast=1,method=DBA_ALL_METHODS)
```

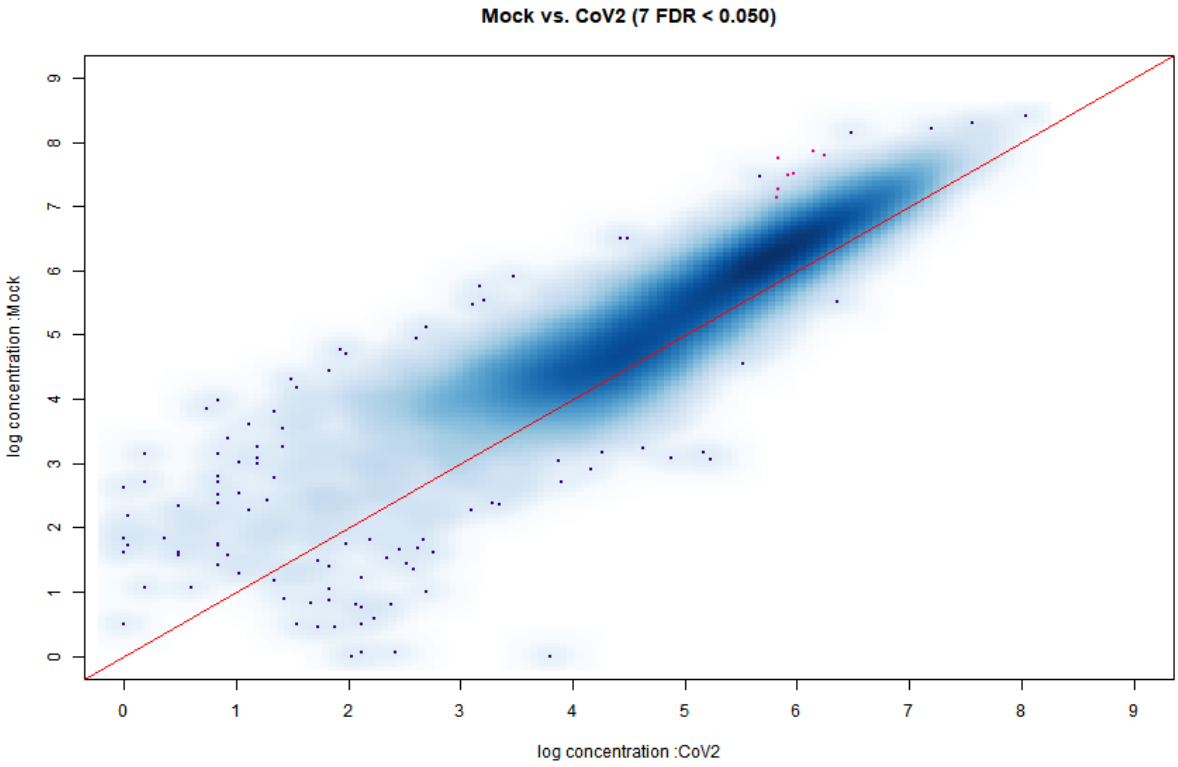



```
dba.plotMA(dbObj, method=DBA_DESEQ2)
```



There is around 4-fold change for mock treatments compared to cov2 treatments in terms of change in binding affinity for significantly differentially bound sites.

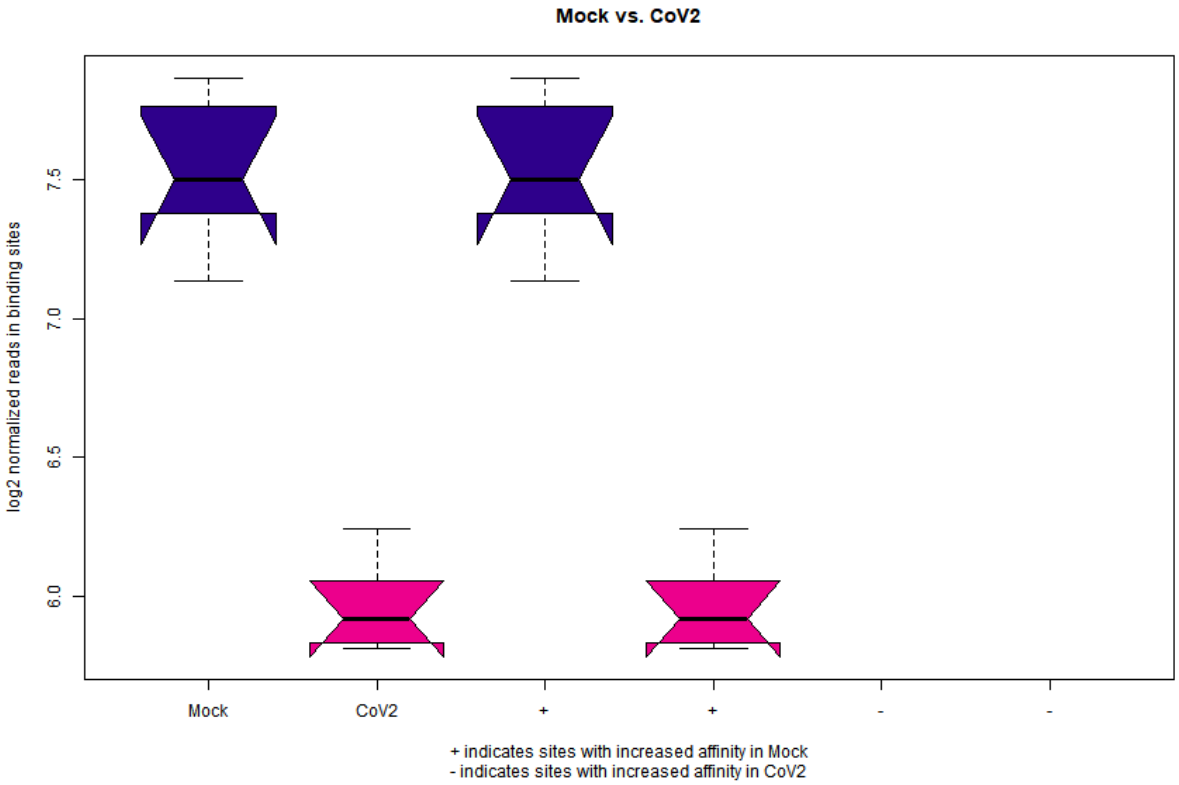
```
dba.plotMA(dbObj, bXY=TRUE)
```



The plot shows the same result as the previous one.

```
pvals <- dba.plotBox(dbObj)
```

```
## Error in wilcox.test.default(toplot[[i]], toplot[[j]], paired = FALSE): 'y'
```



As seen from the MA plots, in the box plots, we also see that there is increased binding for the mock treatments and no increase of binding for Cov2 treatments.

```
res_deseq <- dba.report(dbObj, method=DBA_DESEQ2, contrast = 1, th=1)
res_deseq
```

```
## GRanges object with 19115 ranges and 6 metadata columns:
##         seqnames              ranges strand |      Conc Conc_Mock Conc_CoV2
##            <Rle>           <IRanges>  <Rle> | <numeric> <numeric> <numeric>
##    2405       10   72274718-72275118      * |   7.23945   7.74046   5.83281
##   11804       20   50191873-50192273      * |   7.06196   7.49813   5.97419
##    3622       11   76768859-76769259      * |   7.39195   7.86357   6.14041
##    9844       19   48965610-48966010      * |   7.03710   7.48007   5.91962
##   11371       20       381322-381722      * |   7.34799   7.78773   6.24470
##     ...      ...                 ...    ... .       ...       ...       ...
##    8857       18   57630126-57630526      * |   5.47660   5.49036   5.45571
##   13121        3 124835549-124835949      * |   4.38698   4.36897   4.41358
##    4157       12   46664989-46665389      * |   4.45864   4.46255   4.45276
##    6973       16   28974836-28975236      * |   6.61404   6.61794   6.60816
##    2316       10   60778938-60779338      * |   6.47635   6.47420   6.47957
##                Fold     p-value        FDR
##           <numeric>   <numeric>  <numeric>
##    2405     1.90765 5.28809e-07 0.00070173
##   11804     1.52394 3.99244e-05 0.02127713
##    3622     1.72316 4.81020e-05 0.02127713
##    9844     1.56046 1.44265e-04 0.03521911
```

```
##   11371     1.54303 1.57677e-04 0.03521911
##     ...         ...         ...        ...
##    8857  0.03464552    0.999802          1
##   13121 -0.04460903    0.999859          1
##    4157  0.00978327    0.999904          1
##    6973  0.00978077    0.999906          1
##    2316 -0.00536710    0.999928          1
##   -------
##   seqinfo: 73 sequences from an unspecified genome; no seqlengths
```

```
dim(res_deseq)
```

```
## NULL
```

```
dim(res_deseq[res_deseq$FDR < 0.05])
```

```
## NULL
```

```
out <- as.data.frame(res_deseq)
write.table(out, file="result/MockvsCov_deseq2.txt", sep="\t", quote=F, row.names=F)
```

```
head(out)
```

```
##       seqnames     start       end width strand     Conc Conc_Mock Conc_CoV2
## 2405        10  72274718  72275118   401      * 7.239450  7.740463  5.832812
## 11804       20  50191873  50192273   401      * 7.061955  7.498125  5.974187
## 3622        11  76768859  76769259   401      * 7.391953  7.863575  6.140412
## 9844        19  48965610  48966010   401      * 7.037099  7.480071  5.919615
## 11371       20    381322    381722   401      * 7.347990  7.787734  6.244700
## 15070        5 172770188 172770588   401      * 6.854020  7.275040  5.829838
##           Fold      p.value          FDR
## 2405  1.907651 5.288091e-07 0.0007017297
## 11804 1.523938 3.992443e-05 0.0212771330
## 3622  1.723163 4.810203e-05 0.0212771330
## 9844  1.560456 1.442650e-04 0.0352191146
## 11371 1.543034 1.576774e-04 0.0352191146
## 15070 1.445202 1.719370e-04 0.0352191146
```

```
# Create bed files for each keeping only significant peaks (p < 0.05)
mock_enrich <- out %>% filter(Fold > 0)
mock_write <- mock_enrich %>%
  filter(FDR < 0.05) %>% dplyr::select(seqnames, start, end)
head(mock_write)
```

```
##       seqnames     start       end
## 2405        10  72274718  72275118
## 11804       20  50191873  50192273
## 3622        11  76768859  76769259
## 9844        19  48965610  48966010
## 11371       20    381322    381722
## 15070        5 172770188 172770588
```

```
# Write to file
write.table(mock_write, file="result/Mock_enriched.bed", sep="\t", quote=F,
            row.names=F, col.names=F)
# Create bed files for each keeping only significant peaks (p < 0.05)
cov_enrich <- out %>% filter(Fold < 0)
cov_write <- cov_enrich %>%
  filter(FDR < 0.05) %>%
  dplyr::select(seqnames, start, end)
head(cov_write)
```

```
## [1] seqnames start    end
## <0  > <  row.names    0  >
```

5.7 Find genes of interest

The humangenes object was created in the previous section.

```
mock_enrich$seqnames <- paste0("chr",mock_enrich$seqnames)
cov_enrich$seqnames <- paste0("chr",cov_enrich$seqnames)
#GRanges object of the mock treatment and cov2 treatment
mock_result <- makeGRangesFromDataFrame(mock_enrich)
cov_result <- makeGRangesFromDataFrame(cov_enrich)
# GRanges object with FDR below 0.05
mock_result_sig <- makeGRangesFromDataFrame(mock_enrich %>% filter(FDR < 0.05))
cov_result_sig <- makeGRangesFromDataFrame(cov_enrich %>% filter(FDR < 0.05))
```

```
# Get overlaps
mock_result_genes <- subsetByOverlaps(humangenes,mock_result,ignore.strand=T)
head(mock_result_genes)
```

```
## GRanges object with 6 ranges and 1 metadata column:
##             seqnames              ranges strand |     gene_id
##                <Rle>           <IRanges>  <Rle> | <character>
##           1    chr19   58345178-58362751      - |           1
##         100    chr20   44619522-44652233      - |         100
##   100009676     chr3 101676475-101679217      + |   100009676
##       10002    chr15   71792638-71818259      + |       10002
##   100049716    chr12       630858-664196      + |   100049716
##       10005    chr20   45841721-45857405      - |       10005
##   -------
##   seqinfo: 640 sequences (1 circular) from hg38 genome
```

```
cov_result_genes <- subsetByOverlaps(humangenes,cov_result,ignore.strand=T)
head(cov_result_genes)
```

```
## GRanges object with 6 ranges and 1 metadata column:
##             seqnames              ranges strand |     gene_id
##                <Rle>           <IRanges>  <Rle> | <character>
##       10001    chr14   70581257-70641204      - |       10001
##   100128108    chr15 101737099-101746870      - |   100128108
```

```
##   100128285    chr15   75727670-75738629      - |   100128285
##   100128537     chr1 207801518-207879096      - |   100128537
##   100128714    chr15   25902119-26053123      + |   100128714
##   100128782     chr9   95813170-95876049      - |   100128782
##   -------
##   seqinfo: 640 sequences (1 circular) from hg38 genome
```

```
mock_result_genes_sig <- subsetByOverlaps(humangenes,mock_result_sig,ignore.strand=T)
head(mock_result_genes_sig)
```

```
## GRanges object with 6 ranges and 1 metadata column:
##             seqnames              ranges strand |     gene_id
##                <Rle>           <IRanges>  <Rle> | <character>
##        1051    chr20   50190830-50192668      + |        1051
##   105377730     chr5 172762980-172782334      + |   105377730
##        1843     chr5 172768096-172771195      - |        1843
##        2512    chr19   48965309-48966879      + |        2512
##       54541    chr10   72273919-72276036      + |       54541
##       57761    chr20       362835-397559      + |       57761
##   -------
##   seqinfo: 640 sequences (1 circular) from hg38 genome
```

```
cov_result_genes_sig <- subsetByOverlaps(humangenes,cov_result_sig,ignore.strand=T)
head(cov_result_genes_sig)
```

```
## GRanges object with 0 ranges and 1 metadata column:
##    seqnames    ranges strand |     gene_id
##       <Rle> <IRanges>  <Rle> | <character>
##   -------
##   seqinfo: 640 sequences (1 circular) from hg38 genome
```

```
# Gene annotation
mock_result_genes <- select(org.Hs.eg.db, mock_result_genes$gene_id,
                            c("SYMBOL", "GENENAME"))
colnames(mock_result_genes) <- c("Entrez_ID","Gene_Symbol","Gene_Name")
head(mock_result_genes)
```

```
##   Entrez_ID Gene_Symbol                                    Gene_Name
## 1         1        A1BG                      alpha-1-B glycoprotein
## 2       100         ADA                         adenosine deaminase
## 3 100009676  ZBTB11-AS1                      ZBTB11 antisense RNA 1
## 4     10002       NR2E3 nuclear receptor subfamily 2 group E member 3
## 5 100049716   NINJ2-AS1                       NINJ2 antisense RNA 1
## 6     10005       ACOT8                     acyl-CoA thioesterase 8
```

```
cov_result_genes <- select(org.Hs.eg.db, cov_result_genes$gene_id,
                           c("SYMBOL", "GENENAME"))
colnames(cov_result_genes) <- c("Entrez_ID","Gene_Symbol","Gene_Name")
head(cov_result_genes)
```

```
##   Entrez_ID Gene_Symbol                                    Gene_Name
```

```
## 1     10001        MED6                  mediator complex subunit 6
## 2 100128108   UBE2Q2P13                       UBE2Q2 pseudogene 13
## 3 100128285     DNM1P35                     dynamin 1 pseudogene 35
## 4 100128537  MIR29B2CHG                MIR29B2 and MIR29C host gene
## 5 100128714   LINC02346 long intergenic non-protein coding RNA 2346
## 6 100128782 ERCC6L2-AS1                     ERCC6L2 antisense RNA 1
```

```
mock_result_genes_sig <- select(org.Hs.eg.db, mock_result_genes_sig$gene_id,
                                c("SYMBOL", "GENENAME"))
colnames(mock_result_genes_sig) <- c("Entrez_ID","Gene_Symbol","Gene_Name")
head(mock_result_genes_sig)
```

```
##   Entrez_ID  Gene_Symbol                           Gene_Name
## 1      1051        CEBPB CCAAT enhancer binding protein beta
## 2 105377730 LOC105377730        uncharacterized LOC105377730
## 3      1843        DUSP1      dual specificity phosphatase 1
## 4      2512          FTL                ferritin light chain
## 5     54541        DDIT4   DNA damage inducible transcript 4
## 6     57761        TRIB3             tribbles pseudokinase 3
```

```
cov_result_genes_sig <- select(org.Hs.eg.db, cov_result_genes_sig$gene_id,
                               c("SYMBOL", "GENENAME"))
colnames(cov_result_genes_sig) <- c("Entrez_ID","Gene_Symbol","Gene_Name")
head(cov_result_genes_sig)
```

```
## [1] Entrez_ID   Gene_Symbol Gene_Name
## <0  > <  row.names    0  >
```

```
# Write results to tables
write.table(mock_result_genes,file="result/mock_result_genes.txt",col.names = T,
            row.names = F,quote = F,sep="\t")
write.table(mock_result_genes_sig,file="result/mock_result_genes_sig.txt",
            col.names = T,row.names = F,quote = F,sep="\t")
```

```
# Find genes that are different between mock and control
different <- setdiff(cov_result_genes$Gene_Symbol, mock_result_genes$Gene_Symbol)
head(different)
```

```
## [1] "MED6"        "UBE2Q2P13"   "DNM1P35"     "LINC02346"   "ERCC6L2-AS1"
## [6] "PKP4-AS1"
```

```
length(different)
```

```
## [1] 829
```

5.8 Visualization

```r
# Subset those with correct chromosome names
mock_resultSUB <- mock_enrich[mock_enrich$seqnames
                              %in% paste0("chr", c(1:22, "X", "Y")),]
covid_resultSUB <- cov_enrich[cov_enrich$seqnames
                              %in% paste0("chr", c(1:22, "X", "Y")),]
```

```r
# Make track files
write('track name="Mock" description="Mock treatment" visibility=3 db=hg38',
      file = "result/MockDE_track.broadPeak")
write.table(mock_resultSUB[c("seqnames", "start", 'end')],
            file = "result/MockDE_track.broadPeak", append=T,
            sep = "\t", quote=F, row.names=F, col.names=F)
write('track name="Covid" description="Covid treatment" visibility=3 db=hg38',
      file = "result/CovidDE_track.broadPeak")
write.table(covid_resultSUB[c("seqnames", "start", 'end')],
            file = "result/CovidDE_track.broadPeak", append=T,
            sep = "\t", quote=F, row.names=F, col.names=F)
```

E Session Info

The following section outlines the R package versions.

```
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19044)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=Korean_Korea.utf8  LC_CTYPE=Korean_Korea.utf8
## [3] LC_MONETARY=Korean_Korea.utf8 LC_NUMERIC=C
## [5] LC_TIME=Korean_Korea.utf8
##
## attached base packages:
##  [1] parallel  grid      stats4    stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] ChAMPdata_2.28.0
##  [2] wateRmelon_2.2.0
##  [3] illuminaio_0.38.0
##  [4] IlluminaHumanMethylation450kanno.ilmn12.hg19_0.6.1
##  [5] ROC_1.72.0
##  [6] lumi_2.48.0
##  [7] methylumi_2.42.0
##  [8] minfi_1.42.0
##  [9] bumphunter_1.38.0
## [10] locfit_1.5-9.6
## [11] iterators_1.0.14
## [12] foreach_1.5.2
## [13] Biostrings_2.64.1
## [14] XVector_0.36.0
## [15] FDb.InfiniumMethylation.hg19_2.2.0
## [16] TxDb.Hsapiens.UCSC.hg19.knownGene_3.2.2
## [17] ggplot2_3.3.6
## [18] reshape2_1.4.4
## [19] scales_1.2.1
## [20] VennDiagram_1.7.3
## [21] futile.logger_1.4.3
## [22] DiffBind_3.6.5
## [23] org.Hs.eg.db_3.15.0
## [24] TxDb.Hsapiens.UCSC.hg38.knownGene_3.15.0
## [25] GenomicFeatures_1.48.4
## [26] AnnotationDbi_1.58.0
## [27] GEOquery_2.64.2
## [28] WebGestaltR_0.4.4
## [29] msqrob2_1.4.0
## [30] QFeatures_1.6.0
## [31] MultiAssayExperiment_1.22.0
## [32] SummarizedExperiment_1.26.1
## [33] Biobase_2.56.0
## [34] GenomicRanges_1.48.0
## [35] GenomeInfoDb_1.32.4
```

```
## [36] IRanges_2.30.1
## [37] S4Vectors_0.34.0
## [38] BiocGenerics_0.42.0
## [39] MatrixGenerics_1.8.1
## [40] matrixStats_0.62.0
## [41] edgeR_3.38.4
## [42] limma_3.52.4
## [43] tximport_1.24.0
## [44] dplyr_1.0.10
## [45] biomaRt_2.52.0
##
## loaded via a namespace (and not attached):
##   [1] rappdirs_0.3.3            rtracklayer_1.56.1
##   [3] coda_0.19-4               tidyr_1.2.1
##   [5] bit64_4.0.5               knitr_1.40
##   [7] irlba_2.3.5.1             DelayedArray_0.22.0
##   [9] data.table_1.14.4         hwriter_1.3.2.1
##  [11] KEGGREST_1.36.3           RCurl_1.98-1.9
##  [13] AnnotationFilter_1.20.0   doParallel_1.0.17
##  [15] generics_0.1.3            preprocessCore_1.58.0
##  [17] lambda.r_1.2.4            RSQLite_2.2.18
##  [19] bit_4.0.4                 tzdb_0.3.0
##  [21] xml2_1.3.3                assertthat_0.2.1
##  [23] amap_0.8-19               apeglm_1.18.0
##  [25] xfun_0.33                 hms_1.1.2
##  [27] evaluate_0.17             fansi_1.0.3
##  [29] restfulr_0.0.15           scrime_1.3.5
##  [31] progress_1.2.2            caTools_1.18.2
##  [33] dbplyr_2.2.1              igraph_1.3.5
##  [35] DBI_1.1.3                 htmlwidgets_1.5.4
##  [37] reshape_0.8.9             apcluster_1.4.10
##  [39] purrr_0.3.5               ellipsis_0.3.2
##  [41] annotate_1.74.0           deldir_1.0-6
##  [43] sparseMatrixStats_1.8.0   vctrs_0.5.0
##  [45] cachem_1.0.6              withr_2.5.0
##  [47] BSgenome_1.64.0           bdsmatrix_1.3-6
##  [49] GenomicAlignments_1.32.1  prettyunits_1.1.1
##  [51] mclust_6.0.0              svglite_2.1.0
##  [53] cluster_2.1.3             lazyeval_0.2.2
##  [55] crayon_1.5.2              genefilter_1.78.0
##  [57] pkgconfig_2.0.3           nlme_3.1-157
##  [59] ProtGenerics_1.28.0       rlang_1.0.6
##  [61] lifecycle_1.0.3           nleqslv_3.3.3
##  [63] filelock_1.0.2            affyio_1.66.0
##  [65] BiocFileCache_2.4.0       invgamma_1.1
##  [67] rngtools_1.5.2            base64_2.0.1
##  [69] Matrix_1.5-1              ashr_2.2-54
##  [71] Rhdf5lib_1.18.2           boot_1.3-28
##  [73] whisker_0.4               png_0.1-7
##  [75] rjson_0.2.21              bitops_1.0-7
##  [77] KernSmooth_2.23-20        rhdf5filters_1.8.0
##  [79] blob_1.2.3                DelayedMatrixStats_1.18.2
##  [81] doRNG_1.8.2               mixsqp_0.3-48
##  [83] stringr_1.4.1             SQUAREM_2021.1
```

```
##  [85] nor1mix_1.3-0            ShortRead_1.54.0
##  [87] readr_2.1.3              jpeg_0.1-9
##  [89] memoise_2.0.1            magrittr_2.0.3
##  [91] plyr_1.8.7               gplots_3.1.3
##  [93] zlibbioc_1.42.0          compiler_4.2.1
##  [95] BiocIO_1.6.0             bbmle_1.0.25
##  [97] RColorBrewer_1.1-3       clue_0.3-62
##  [99] lme4_1.1-31              Rsamtools_2.12.0
## [101] cli_3.4.1                affy_1.74.0
## [103] systemPipeR_2.2.2        formatR_1.12
## [105] MASS_7.3-58.1            mgcv_1.8-40
## [107] tidyselect_1.2.0         stringi_1.7.8
## [109] emdbook_1.3.12           yaml_2.3.5
## [111] askpass_1.1              latticeExtra_0.6-30
## [113] ggrepel_0.9.1            tools_4.2.1
## [115] rstudioapi_0.14          MsCoreUtils_1.8.0
## [117] BiocManager_1.30.19      digest_0.6.29
## [119] quadprog_1.5-8           Rcpp_1.0.9
## [121] siggenes_1.70.0          httr_1.4.4
## [123] colorspace_2.0-3         XML_3.99-0.12
## [125] truncnorm_1.0-8          splines_4.2.1
## [127] multtest_2.52.0          systemfonts_1.0.4
## [129] xtable_1.8-4             jsonlite_1.8.3
## [131] nloptr_2.0.3             futile.options_1.0.1
## [133] R6_2.5.1                 pillar_1.8.1
## [135] htmltools_0.5.3          glue_1.6.2
## [137] fastmap_1.1.0            minqa_1.2.5
## [139] BiocParallel_1.30.4      beanplot_1.3.1
## [141] codetools_0.2-18         GreyListChIP_1.28.1
## [143] mvtnorm_1.1-3            utf8_1.2.2
## [145] lattice_0.20-45          tibble_3.1.8
## [147] numDeriv_2016.8-1.1      curl_4.3.3
## [149] gtools_3.9.4             openssl_2.0.4
## [151] interp_1.1-3             survival_3.3-1
## [153] rmarkdown_2.17           munsell_0.5.0
## [155] rhdf5_2.40.0             GenomeInfoDbData_1.2.8
## [157] HDF5Array_1.24.2         gtable_0.3.1
```