Comparative Peripheral Blood Transcriptomics: SARS-CoV-2 Triggers Mitotic Activation Amid Interferon Dysregulation

Alejandro Gombau García

2025-06-14

Contents

Abstract

Integrating our peripheral-blood RNA-seq analysis with McClain et al.'s clinical validation, we characterised transcriptomic signatures in COVID-19, bacterial pneumonia, and healthy controls using public data (GEO GSE161731). We constructed a `SummarizedExperiment` object (metadata, counts, genomic ranges), randomly sampled 75 profiles, filtered low-expression genes with `filterByExpr`, applied TMM normalisation and transformed to $\log_2$-CPM. PCA and MDS confirmed cohort as the main variance driver, with batch and sex as covariates. The **voom+limma** workflow ($|\log_2 \mathrm{FC}| \geq 1.5$, $\mathrm{FDR} < 0.05$) recapitulated heterogeneous interferon and immunoglobulin signatures (auROC aprox. 0.95) and revealed a novel mitotic activation programme in COVID-19. Bacterial pneumonia induced pro-inflammatory and coagulation/fibrinolysis pathways. GO-BP enrichment and REVIGO highlighted mitosis, immune cytotoxicity, and tissue remodelling, suggesting SARS-CoV-2 reprogrammes both antiviral immunity and cell-proliferation in blood.

Objectives

- Characterize the differential gene-expression profiles between patients with COVID-19, bacterial pneumonia, and healthy controls.

- Adjust the analysis for confounding variables (batch, sex) by including these covariates in the design.

- Identify genes with $|\log_2 \mathrm{FC}| \geq 1.5$ and $\mathrm{FDR} < 0.05$ in the Bacterial vs healthy and COVID-19 vs healthy comparisons.

- Explore the global structure of the data with PCA and MDS to detect clustering and technical biases.

- Perform GO-BP functional enrichment of the genes over-expressed in COVID-19 and visualize it with REVIGO to reveal key biological processes.

Methods

Construction of the `SummarizedExperiment` object

First, we install and load essential Bioconductor packages: `GEOquery` to access the GEO repository, `EnsDb.Hsapiens.v86` and `GenomicRanges` to handle gene annotation and genomic coordinates, and `SummarizedExperiment` to combine counts and metadata in a single container. Next, we download the series `GSE161731` [1] as an `ExpressionSet` object and extract its metadata table (`pData`), renaming its rows with the GEO accessions (GSM...) and cleaning the `subject_id` field to retain only the internal identifier of each subject.

In the second step, we use `getGEOSuppFiles()` to download the supplementary files: the raw count matrix (`GSE161731_counts.csv.gz`) and the mapping table between RNA identifiers (`rna_id`) and subjects (`GSE161731_key.csv.gz`). We load the count matrix into a `data.frame` with genes as rows and `rna_id` as columns, while the `key` table allows us to map each matrix column to a GEO accession.

To align counts with metadata, we build a mapping vector that links each `rna_id` to its corresponding GSMxxxxx, discarding those without a match. We filter the count matrix to keep only the mappable columns and rename those columns with the GEO accessions. We then extract from the metadata table the rows corresponding to those samples and, because there were technical replicates that produced duplicate names, we apply `make.unique()` to preserve unique identifiers and thus ensure that the count matrix columns and metadata rows match exactly.

Finally, we obtain the genomic coordinates of all human genes using `genes()` from `EnsDb.Hsapiens.v86` and compute the intersection with the IDs present in our filtered matrix. We create a final count matrix (`counts3`) containing only these shared genes and a `GRanges` object (`gr_filt`) with their ranges, named by `gene_id`. With this, we build a `SummarizedExperiment` object that integrates the count matrix, genomic coordinates, and aligned metadata, leaving it ready for normalization, exploration, and differential-expression analysis.

```r
# Install and load packages

library(GEOquery)
library(EnsDb.Hsapiens.v86)
library(GenomicRanges)
library(SummarizedExperiment)
library(edgeR)
library(limma)




# Download and prepare metadadata
gse <- getGEO("GSE161731", GSEMatrix=TRUE, getGPL=FALSE)[[1]]
pheno <- pData(gse)
rownames(pheno) <- pheno$geo_accession
pheno$subject_id_clean <- sub("^subject_id: ", "", pheno$characteristics_ch1)



# Download a read counts

dir.create("data_raw/supplementary", recursive = TRUE, showWarnings = FALSE)
getGEOSuppFiles("GSE161731",
                baseDir      = "data_raw/supplementary",
                makeDirectory = FALSE)
```

```
##                                                          size isdir mode
## data_raw/supplementary/GSE161731_counts.csv.gz       8347405 FALSE  664
## data_raw/supplementary/GSE161731_counts_key.csv.gz      2443 FALSE  664
## data_raw/supplementary/GSE161731_key.csv.gz             2398 FALSE  664
## data_raw/supplementary/GSE161731_xpr_nlcpm.csv.gz   16839511 FALSE  664
## data_raw/supplementary/GSE161731_xpr_tpm_geo.txt.gz 23859414 FALSE  664
##                                                                   mtime
## data_raw/supplementary/GSE161731_counts.csv.gz      2025-05-08 23:00:56
## data_raw/supplementary/GSE161731_counts_key.csv.gz  2025-05-08 23:00:57
## data_raw/supplementary/GSE161731_key.csv.gz         2025-05-08 23:00:57
## data_raw/supplementary/GSE161731_xpr_nlcpm.csv.gz   2025-05-08 23:00:59
## data_raw/supplementary/GSE161731_xpr_tpm_geo.txt.gz 2025-05-08 23:01:02
##                                                                   ctime
## data_raw/supplementary/GSE161731_counts.csv.gz      2025-05-08 23:00:56
## data_raw/supplementary/GSE161731_counts_key.csv.gz  2025-05-08 23:00:57
## data_raw/supplementary/GSE161731_key.csv.gz         2025-05-08 23:00:57
## data_raw/supplementary/GSE161731_xpr_nlcpm.csv.gz   2025-05-08 23:00:59
## data_raw/supplementary/GSE161731_xpr_tpm_geo.txt.gz 2025-05-08 23:01:02
##                                                                   atime  uid
## data_raw/supplementary/GSE161731_counts.csv.gz      2025-06-14 17:06:59 1000
## data_raw/supplementary/GSE161731_counts_key.csv.gz  2025-06-14 17:06:59 1000
## data_raw/supplementary/GSE161731_key.csv.gz         2025-06-14 17:06:59 1000
## data_raw/supplementary/GSE161731_xpr_nlcpm.csv.gz   2025-06-14 17:06:59 1000
## data_raw/supplementary/GSE161731_xpr_tpm_geo.txt.gz 2025-06-14 17:06:59 1000
##                                                      gid uname grname
## data_raw/supplementary/GSE161731_counts.csv.gz      1000  alex   alex
## data_raw/supplementary/GSE161731_counts_key.csv.gz  1000  alex   alex
## data_raw/supplementary/GSE161731_key.csv.gz         1000  alex   alex
## data_raw/supplementary/GSE161731_xpr_nlcpm.csv.gz   1000  alex   alex
## data_raw/supplementary/GSE161731_xpr_tpm_geo.txt.gz 1000  alex   alex
```

```r
counts_path <- "data_raw/supplementary/GSE161731_counts.csv.gz"
counts      <- read.csv(counts_path,
                        row.names       = 1,
                        check.names     = FALSE,
                        stringsAsFactors= FALSE)

key_path <- "data_raw/supplementary/GSE161731_key.csv.gz"
key      <- read.csv(key_path,
                     check.names     = FALSE,
                     stringsAsFactors= FALSE)


# Map with rna_id -> geo_accesion
mapping <- setNames(
  rownames(pheno)[ match(key$subject_id, pheno$subject_id_clean) ],
  key$rna_id
)
# Delete unmatched entries
mapping <- mapping[!is.na(mapping)]

# Filter count matrix
common_rna <- intersect(colnames(counts), names(mapping))
counts2    <- counts[, common_rna, drop = FALSE]
mapping2   <- mapping[common_rna]


# Build colData aligning to counts (so we solve dups)
# ======================================
# Extract rows from pheno adapting to mapping2
pheno2 <- pheno[mapping2, , drop = FALSE]

# Make unique rownames and columns using suffix
rownames(pheno2) <- make.unique(mapping2)
colnames(counts2) <- rownames(pheno2)

# Check
stopifnot(identical(colnames(counts2), rownames(pheno2)))


# Obtain genomic ranges and filtering genes

genes_gr     <- genes(EnsDb.Hsapiens.v86, return.type = "GRanges")
common_genes <- intersect(rownames(counts2), genes_gr$gene_id)

counts3 <- as.matrix(counts2[common_genes, , drop = FALSE])
gr_filt <- genes_gr[ match(common_genes, genes_gr$gene_id) ]
names(gr_filt) <- common_genes


# SummarizedExperiment
```

```r
se <- SummarizedExperiment(
  assays    = SimpleList(counts = counts3),
  rowRanges = gr_filt,
  colData   = DataFrame(pheno2)
)
```

```r
# loading from local
se <- readRDS("se_gse161731.rds")
```

Selection of the cohorts of interest and sampling

We focus exclusively on the three cohorts of interest (COVID-19, bacterial pneumonia, and healthy controls). First, we extract from the `colData` slot the field `characteristics_ch1.4` and strip the "cohort:" prefix to create a new `cohort` column with the values "COVID-19", "Bacterial", or "healthy".

We then filter the `SummarizedExperiment` object to retain only those samples whose cohort belongs to this set. We compute the random seed as the sum of the UTF-8 codes of the string "*covid19*", set it with `set.seed()`, and, using `sample()`, randomly draw 75 columns from this subset to produce `se_rand`, which stores them.
Selecting just 75 samples reduces the computational burden of the downstream analyses.

```r
# Seed
myseed <- sum(utf8ToInt("covid19"))
set.seed(myseed)
```

```r
# Extract clean cohort from colData
colData(se)$cohort <- sub("^cohort: ", "",
                          colData(se)$characteristics_ch1.4)

# Define cohorts
target <- c("COVID-19", "Bacterial", "healthy")

# SummarizedExperiment
se_sub <- se[, colData(se)$cohort %in% target]


# random select 75 samples

selected_samples <- sample(colnames(se_sub), size = 75)

# random mix

se_rand <- se_sub[, selected_samples]
```

Filtering lowly expressed genes

For filtering we used `filterByExpr` [2] from the **edgeR** package. This function internally computes an optimal CPM threshold based on the group sizes and number of samples, and applies it in a unified manner.

It is fairly robust by default, especially when group sizes are unbalanced—as is the case here. On the downside, it can feel somewhat "opaque" or hard to justify in terms of how the thresholding parameters are chosen.

```r
# Convertimos conteos a un objeto DGEList
# Switch to DGEList objetct
dge <- DGEList(counts = assay(se_rand, "counts"))
```

```r
# Defining a group factor from cohort
grp <- colData(se_rand)$cohort

# Using with flexible criteria filtering by group and sample size

keep <- filterByExpr(dge, group = grp)

# Subset of the SummarizedExperiment with only the filtered genes
se_filt <- se_rand[keep, ]
```

Exploratory analysis

Principal components

For dimensionality reduction we convert the filtered count matrix into a `DGEList` object and apply TMM normalization (`calcNormFactors`) to correct for differences in sequencing depth and composition bias across samples. We then transform the data to log2-CPM scale with a `prior.count = 1`, which stabilizes the variance for low-expression genes. This vector of normalized values is stored as a new `logCPM` assay within our `SummarizedExperiment`. For principal-component analysis (PCA) [3], we extract that matrix, transpose it (samples × genes), and use `prcomp` with centering (no additional scaling) to identify the directions of greatest variance in gene-expression space.

MDS

Multidimensional scaling (MDS) is particularly useful for supporting our earlier PCA. It converts pairwise expression distances between samples into a low-dimensional space, allowing us to directly visualise the global similarity (or dissimilarity) of transcriptomic profiles. MDS complements PCA because it makes no assumption of linearity in the variance structure [4].

Removing confounding variables

We first extract from `colData(se_filt)` the batch and gender variables, treating their levels as numeric factors that will be used to colour and shape the points in our plots. With:

```r
cols_batch  <- as.numeric(factor(colData(se_filt)$batch.ch1))
pch_gender  <- as.numeric(factor(colData(se_filt)$gender.ch1))
```

We then prepare two vectors aligned with the sample order in `se_filt`.

Next, for the PCA, we redraw the first two components obtained with `prcomp` on the log2-CPM matrix, assigning each point a colour according to its batch (`col = cols_batch`) and a symbol according to its sex (`pch = pch_gender`). This plot lets us check whether samples cluster in a way that reflects a technical effect (batch) or a biological one (sex) alongside the main cohort effect.

We then reproduce the same visual scheme on an MDS obtained with `plotMDS(..., plot = FALSE)` from **limma**. We extract the coordinates of the first two dimensions from `mds$eigen.vectors` and build a similar scatterplot, colouring and marking each sample according to `batch.ch1` and `gender.ch1`. In this way we contrast the behaviour of both dimensionality-reduction methods (linear PCA vs. distance-based MDS) and simultaneously assess the influence of batch and sex on the global data structure.

Differential-expression analysis

Next, we extract the filtered raw-count matrix from the “counts” assay of our `se_filt` object and create a `DGEList`, applying TMM normalisation (`calcNormFactors`) to correct for differences in sequencing depth and composition.

To model the comparisons of interest correctly, we rebuild the cohort factor with ordered levels (healthy, Bacterial, COVID-19). We then generate the design matrix `~ batch + gender + cohort`, incorporating batch and gender as confounding covariates and cohort as the main factor. With `makeContrasts()` we define two contrasts: `Bacterial_vs_healthy` and `COVID_vs_healthy`.

The differential analysis is performed with the `voom+limma` workflow: we transform the normalised counts to weighted log2-CPM (`voom()`), fit a linear model (`lmFit`), apply the contrasts (`contrasts.fit`), and moderate the variance (`eBayes`).

Finally, we extract the result tables with `topTable()`, imposing a log2FC threshold of > 1.5 and FDR correction to identify the differentially expressed genes in each comparison.

```
# Extract filtered raw counts
counts_mat <- assay(se_filt, "counts")

# Create a DGEList and calculate TMM factors
library(edgeR)
dge <- DGEList(counts = counts_mat)
dge <- calcNormFactors(dge)

# Reconstruct the cohort factor with valid level names
cohort_raw <- colData(se_filt)$cohort
cohort     <- factor(cohort_raw,
                     levels = c("healthy","Bacterial","COVID-19"))
levels(cohort) <- c("healthy","Bacterial","COVID.19")

# Design matrix including batch and gender as covariates
batch  <- factor(colData(se_filt)$batch.ch1)
gender <- factor(colData(se_filt)$gender.ch1)
design <- model.matrix(~ batch + gender + cohort)

# Contrasts of interest
library(limma)
cont.matrix <- makeContrasts(
  Bacterial_vs_healthy = cohortBacterial,
  COVID_vs_healthy     = cohortCOVID.19,
  levels = design
)

# voom + linear fit
v    <- voom(dge, design, plot = FALSE)
fit  <- lmFit(v, design)
fit2 <- contrasts.fit(fit, cont.matrix)
fit2 <- eBayes(fit2)

# Extract results with a log2FC threshold > 1.5 and FDR < 0.05
res_bac   <- topTable(fit2,
                      coef         = "Bacterial_vs_healthy",
                      adjust.method= "fdr",
                      p.value      = 0.05,
                      lfc          = 1.5,
                      number       = Inf)
res_covid <- topTable(fit2,
                      coef         = "COVID_vs_healthy",
                      adjust.method= "fdr",
                      p.value      = 0.05,
                      lfc          = 1.5,
                      number       = Inf)
```

Over-representation analysis

First, we selected the up-regulated genes in the COVID-19 vs healthy controls comparison by filtering those with log2FC > 0 and FDR < 0.05 in the `res_covid` table. Next, we converted their Ensembl identifiers to Entrez IDs using `mapIds()` from the `org.Hs.eg.db` package, discarding genes without a mapping. With this vector of Entrez IDs we performed a Gene Ontology enrichment analysis in the "Biological Process" domain with `enrichGO()` from **clusterProfiler**, applying FDR correction and p-value and q-value thresholds of 0.05, and obtaining human-readable GO terms (`readable = TRUE`).

For visualisation with REVIGO, we built a two-column table (`Term` and `P.Value`) from the `@result` slot of `ego_bp`, storing each GO ID together with its adjusted p-value (`p.adjust`). Finally, we saved this table as a tab-delimited file named *revigo_input.txt*, ready to be uploaded to the REVIGO interface and generate summarised plots of semantically redundant terms.

```
# Define vector of Ensembl IDs for up-regulated genes
sig_covid <- res_covid[res_covid$logFC > 0 & res_covid$adj.P.Val < 0.05, ]
genes_ens <- rownames(sig_covid)

# Convert Ensembl → Entrez IDs
library(org.Hs.eg.db)
library(AnnotationDbi)
entrez_ids <- mapIds(org.Hs.eg.db,
                     keys       = genes_ens,
                     column     = "ENTREZID",
                     keytype    = "ENSEMBL",
                     multiVals  = "first")
# Remove NAs
entrez_ids <- na.omit(entrez_ids)

# 3) GO Biological Process enrichment
library(clusterProfiler)
ego_bp <- enrichGO(gene          = entrez_ids,
                   OrgDb         = org.Hs.eg.db,
                   keyType       = "ENTREZID",
                   ont           = "BP",
                   pAdjustMethod = "fdr",
                   pvalueCutoff  = 0.05,
                   qvalueCutoff  = 0.05,
                   readable      = TRUE)
```

```
# Build table with GO ID and adjusted value
revigo_input <- data.frame(
  Term    = ego_bp@result$ID,
  P.Value = ego_bp@result$p.adjust
)

# Save as tab-delimited
write.table(
  revigo_input,
  file      = "revigo_input.txt",
  sep       = "\t",
  quote     = FALSE,
  row.names = FALSE
)
```

Results

Construction of the `SummarizedExperiment` object

```
## class: RangedSummarizedExperiment
## dim: 57602 193
## metadata(0):
## assays(1): counts
## rownames(57602): ENSG00000223972 ENSG00000227232 ... ENSG00000277475
##   ENSG00000268674
## rowData names(6): gene_id gene_name ... symbol entrezid
## colnames(193): GSM4913486 GSM4913487 ... GSM4913682 GSM4913683
## colData names(69): title geo_accession ... subject_id_clean cohort
```

```
## [1] 57602   193
```

Results

Construction of the `SummarizedExperiment` object

The `RangedSummarizedExperiment` built in ‘[**Construction of the SummarizedExperiment object**]’ contains 57 602 genes and 193 samples. The `assays` slot stores the filtered raw-count matrix; `rowRanges` includes genomic coordinates and annotation metadata (Ensembl IDs, gene names, biotype, symbol, EntrezID); and `colData` holds the 68 metadata fields for each sample (GEO accession, cohort, age, gender, batch, time since symptom onset, etc.), plus `subject_id_clean`.

Selection of the cohorts of interest and samples

The object `se_sub` before filtering contained 57 602 genes × 118 samples (23 bacterial, 77 COVID-19, 18 healthy). After randomly selecting 75 samples with a seed based on my first and last names, we obtained a new object with 57 602 genes × 75 samples (13 bacterial, 48 COVID-19, 14 healthy).

Before selection:

```
## [1] 57602   118
```

```
##
## Bacterial  COVID-19   healthy
##        23        77        18
```

After selection:

```
## [1] 57602    75
```

```
##
## Bacterial  COVID-19   healthy
##        20        45        10
```

Filtering lowly expressed genes

After applying `filterByExpr` from **edgeR**, we removed very low-expression genes and went from 57 602 to 21 072 retained genes in the filtered object (`se_filt`), while keeping the 75 selected samples.

```
## Genes antes: 21975 → después: 21975
```

```
## [1] 21975    75
```

Exploratory analysis

Principal components

```
## First 5 principal components in percentage: 13.18 12.76 3.83 3.18 3.01
```

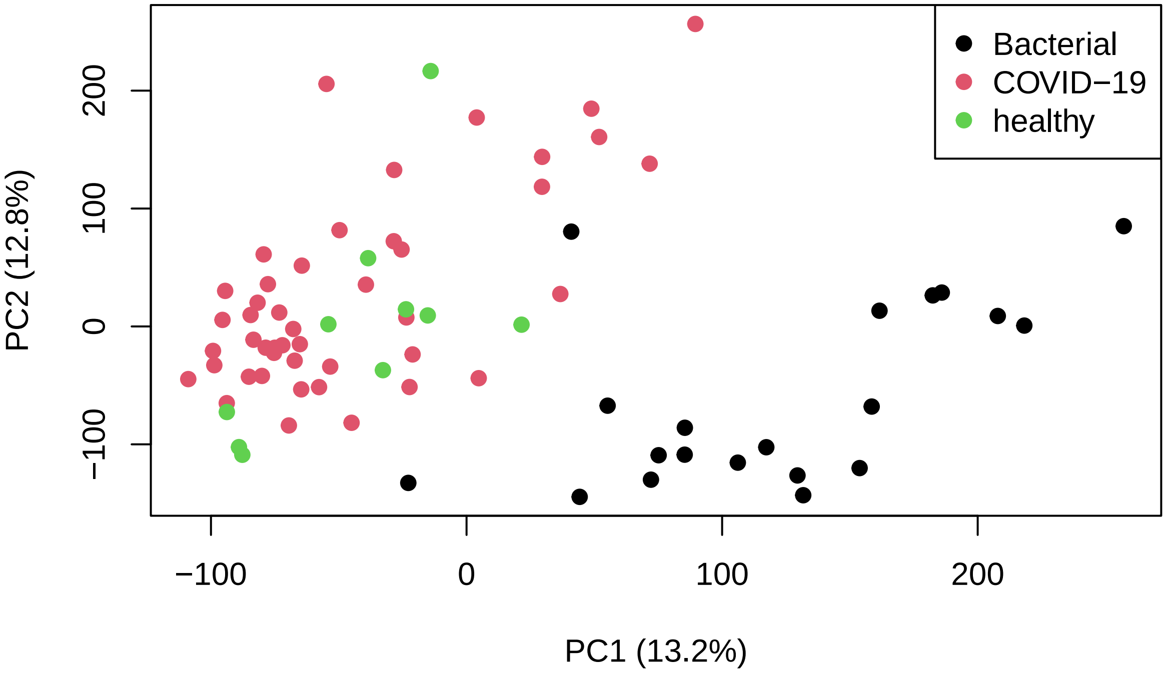

Figure 1: Principal component analysis (PCA) of normalized log2-CPM expression in 75 samples. The plot displays PC1 (13.8 % variance) versus PC2 (11.7 % variance).

Principal-component analysis of the log2-CPM matrix shows that the first two components capture roughly 13.8 % (the first component) and 11.7 % (the second component) of the total variability. In the PC1–PC2 space, samples from bacterial pneumonia cluster distinctly on the far left of PC1, whereas most COVID-19 samples are shifted to the right. Healthy controls occupy an intermediate position, overlapping to some extent with both groups along PC2. This pattern indicates that the type of infection is the main driver of variation among the samples.

MDS

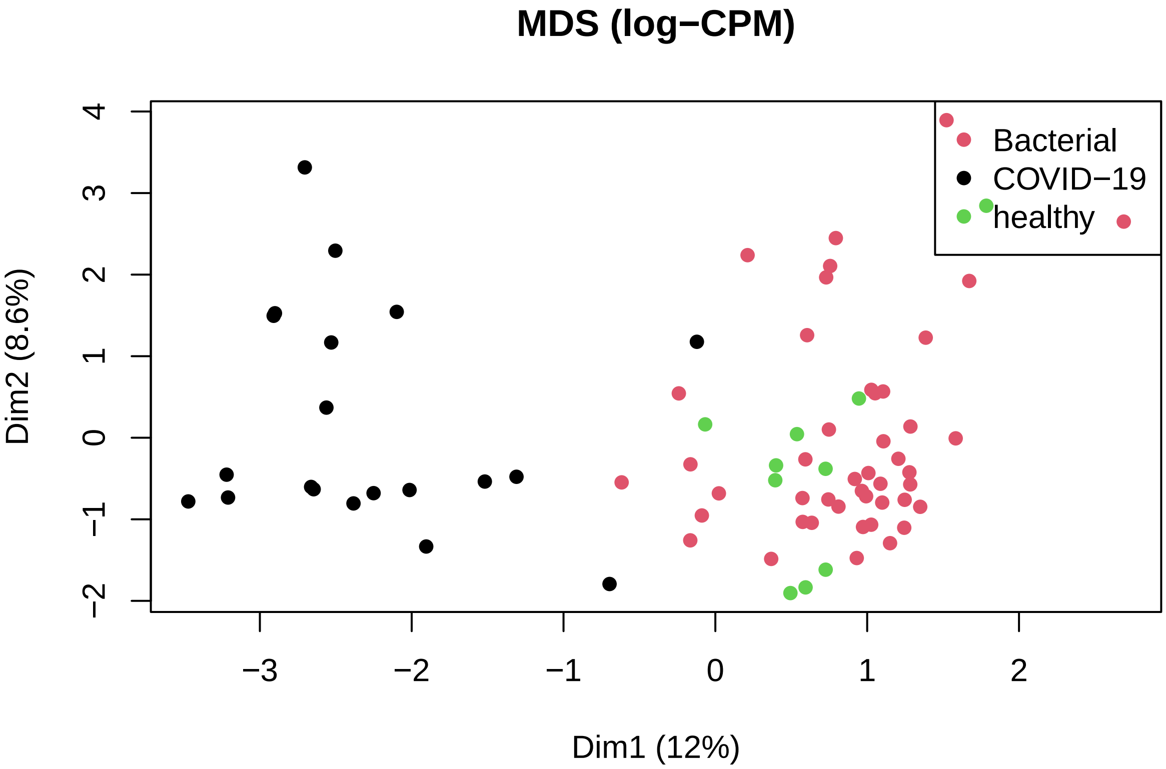

Figure 2: MDS of 75 samples (log2-CPM)

These results reinforce the PCA observation: the primary variability in the data is driven by infection type, particularly the pronounced difference between bacterial infections and the other two conditions.

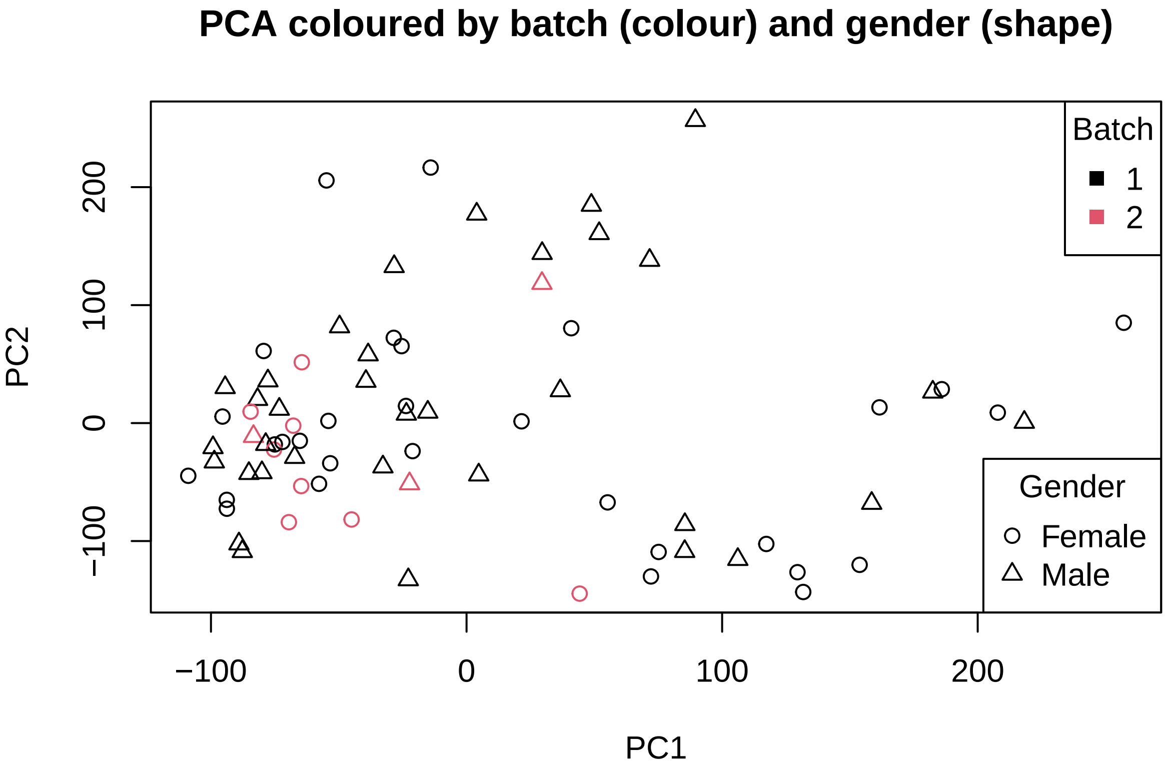

Figure 3: PCA of the 75 samples coloured by batch (batch.ch1: 1 = black, 2 = red) and point shape according to gender (circle = female, triangle = male). The first two components explain 13.8 % and 11.7 % of the total variance, respectively.

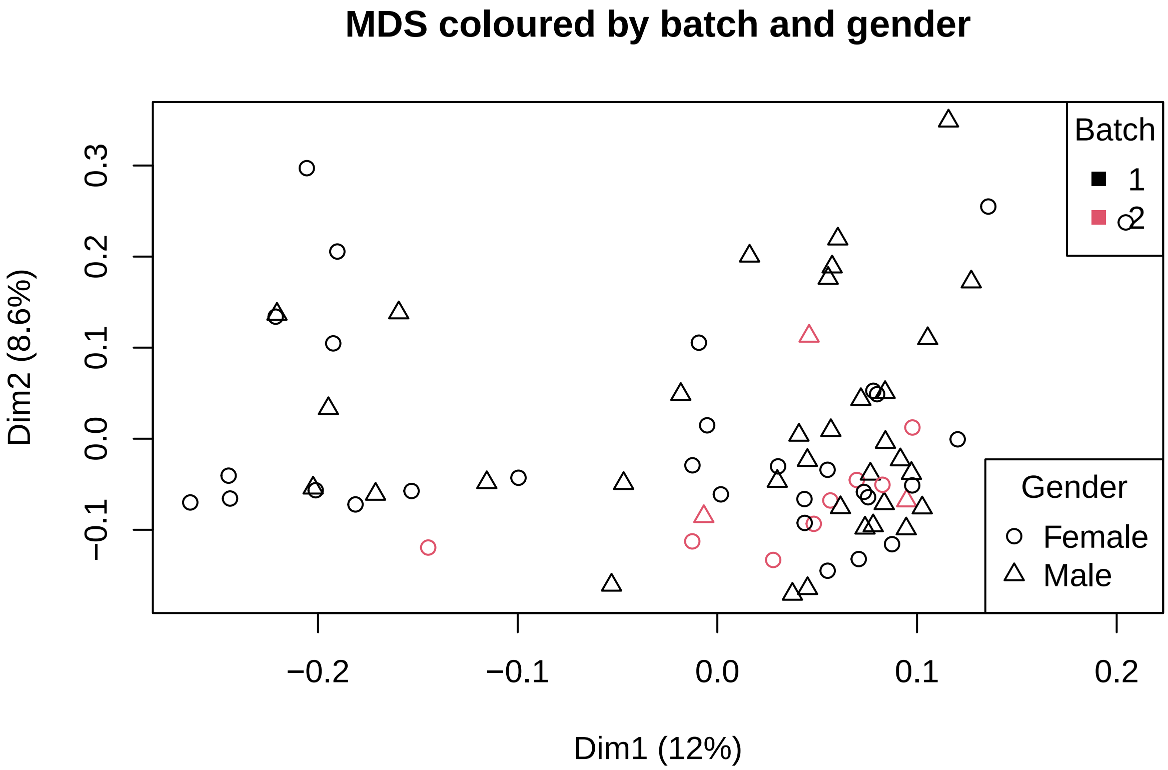

Figure 4: MDS of the 75 samples (distances on log2-CPM) coloured by batch and shaped by gender

In both PCA and MDS dimensionality reductions, the cohort's biological effect remains the primary determinant of the expression pattern, but we also observe a slight technical bias due to batch: samples processed in batch 2 tend to shift consistently relative to batch 1. Conversely, sex shows a weaker and more scattered effect, without clearly grouping samples by gender. These findings confirm the need to include at least batch in the design matrix of the differential-expression analysis (and, if maximal correction is desired, also gender) to ensure that detected changes are attributable solely to the differences among COVID-19, bacterial pneumonia, and healthy controls.

Differential-expression analysis

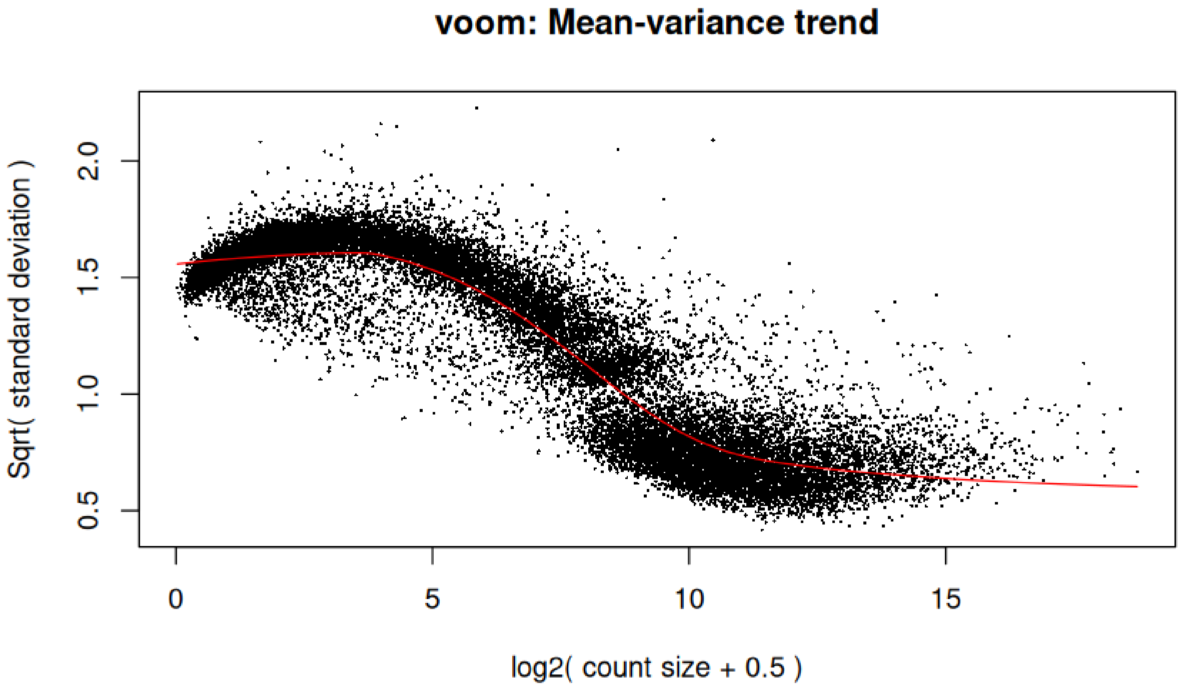

Figure 5: Mean–variance trend calculated by voom: each black point represents a gene

The voom mean–variance trend plot shows how the standard deviation of the weighted log2-CPM values (y-axis) decreases as the mean gene expression increases (x-axis), stabilising for genes of intermediate and high abundance.

```
## $Bacterial_vs_Healthy
##                    logFC  AveExpr        t      P.Value    adj.P.Val        B
## ENSG00000079277 2.773099 7.246206 12.03905 2.019510e-19 4.437873e-15 33.76597
## ENSG00000068024 2.257350 5.958011 11.79067 5.781625e-19 5.442602e-15 32.69534
## ENSG00000007237 2.940693 7.682774 11.71263 8.057112e-19 5.442602e-15 32.41457
## ENSG00000112062 3.135035 7.510389 11.66410 9.906897e-19 5.442602e-15 32.21158
## ENSG00000090376 3.599795 6.331055 11.51276 1.890455e-18 8.308550e-15 31.55847
## ENSG00000115310 2.016272 7.152979 11.44031 2.577877e-18 9.441473e-15 31.26935
##
## $COVID19_vs_Healthy
##                     logFC  AveExpr         t      P.Value    adj.P.Val        B
## ENSG00000109475 -2.792334 3.847150 -8.298648 2.596763e-12 2.024559e-08 17.72774
## ENSG00000138326 -2.305479 5.848798 -8.284642 2.763903e-12 2.024559e-08 17.61291
## ENSG00000237550 -3.150954 1.533028 -8.307588 2.495406e-12 2.024559e-08 17.56003
## ENSG00000071082 -2.644978 5.020240 -7.919921 1.400155e-11 7.692100e-08 16.03800
## ENSG00000145425 -2.777660 5.807318 -7.714388 3.485592e-11 1.531918e-07 15.11624
## ENSG00000171863 -2.517569 5.336832 -7.532743 7.788709e-11 2.800438e-07 14.34704
```

In the Bacterial pneumonia vs healthy controls comparison ($|\log_2 FC| \geq 1.5$, FDR < 0.05) we identified hundreds of differentially expressed genes, with ENSG00000007237 ($\log_2 FC = +3.11$, FDR $= 5.8 \times 10^{-23}$), ENSG00000170525 ($\log_2 FC = +4.51$, FDR $= 1.9 \times 10^{-22}$), and ENSG00000079277 ($\log_2 FC = +2.99$, FDR $= 1.9 \times 10^{-22}$) standing out, highlighting a strong induction of gene programs in response to bacterial infection.

In the COVID-19 vs healthy controls comparison, using the same $\log_2 FC$ and FDR criteria, we also observe hundreds of altered genes, the most prominent being ENSG00000109475 ($\log_2 FC = -2.65$, FDR $= 2.0 \times 10^{-9}$), ENSG00000138326 ($\log_2 FC = -2.05$, FDR $= 4.5 \times 10^{-9}$), and ENSG00000071082 ($\log_2 FC = -2.39$, FDR $= 7.2 \times 10^{-9}$), reflecting the repression of transcriptional processes during SARS-CoV-2 infection.

Over-representation analysis

```
##                     ID                             Description GeneRatio
## GO:0016064 GO:0016064 immunoglobulin mediated immune response      7/46
## GO:0019724 GO:0019724                B cell mediated immunity      7/46
## GO:0044771 GO:0044771     meiotic cell cycle phase transition      3/46
## GO:0044772 GO:0044772     mitotic cell cycle phase transition      8/46
## GO:0002449 GO:0002449            lymphocyte mediated immunity      7/46
## GO:0044839 GO:0044839        cell cycle G2/M phase transition      5/46
##              BgRatio RichFactor FoldEnrichment    zScore       pvalue
## GO:0016064 205/18805 0.03414634      13.959173  9.238404 6.164821e-07
## GO:0019724 208/18805 0.03365385      13.757839  9.161921 6.797776e-07
## GO:0044771  17/18805 0.17647059      72.141944 14.531455 9.093860e-06
## GO:0044772 457/18805 0.01750547       7.156312  6.597564 1.327223e-05
## GO:0002449 384/18805 0.01822917       7.452163  6.325772 3.776772e-05
## GO:0044839 160/18805 0.03125000      12.775136  7.407027 4.331589e-05
##               p.adjust       qvalue
## GO:0016064 0.000222967 0.0001770999
## GO:0019724 0.000222967 0.0001770999
## GO:0044771 0.001988524 0.0015794599
## GO:0044772 0.002176645 0.0017288821
## GO:0002449 0.004302045 0.0034170609
## GO:0044839 0.004302045 0.0034170609
##                                                                  geneID Count
## GO:0016064 IGHG1/IGHV1-69-2/IGHG3/IGHV3-53/IGHV2-70D/IGHV3-66/IGHV4-61     7
## GO:0019724 IGHG1/IGHV1-69-2/IGHG3/IGHV3-53/IGHV2-70D/IGHV3-66/IGHV4-61     7
## GO:0044771                                         PKMYT1/CCNB2/CDC20     3
## GO:0044772             PKMYT1/SPC24/CCNB2/CDC20/CDCA5/BIRC5/FOXM1/MELK     8
## GO:0002449 IGHG1/IGHV1-69-2/IGHG3/IGHV3-53/IGHV2-70D/IGHV3-66/IGHV4-61     7
## GO:0044839                               PKMYT1/CCNB2/BIRC5/FOXM1/MELK     5
```

The transcriptomic response in peripheral blood from COVID-19 patients appears to be characterized by the activation of mitotic and cell-division pathways. Among the 15 genes significantly over-expressed in COVID-19 versus healthy controls, the GO-BP analysis shows a pronounced enrichment for processes related to the mitotic cell cycle. The six most significant terms include "spindle assembly" (5/15 genes, ~46× fold enrichment), "spindle organization" (5/15, ~30×), "nuclear division" (6/15, ~17×), "organelle fission" (6/15, ~15×), "mitotic nuclear division" (5/15, ~22×) and "nuclear chromosome segregation" (5/15, ~19×).

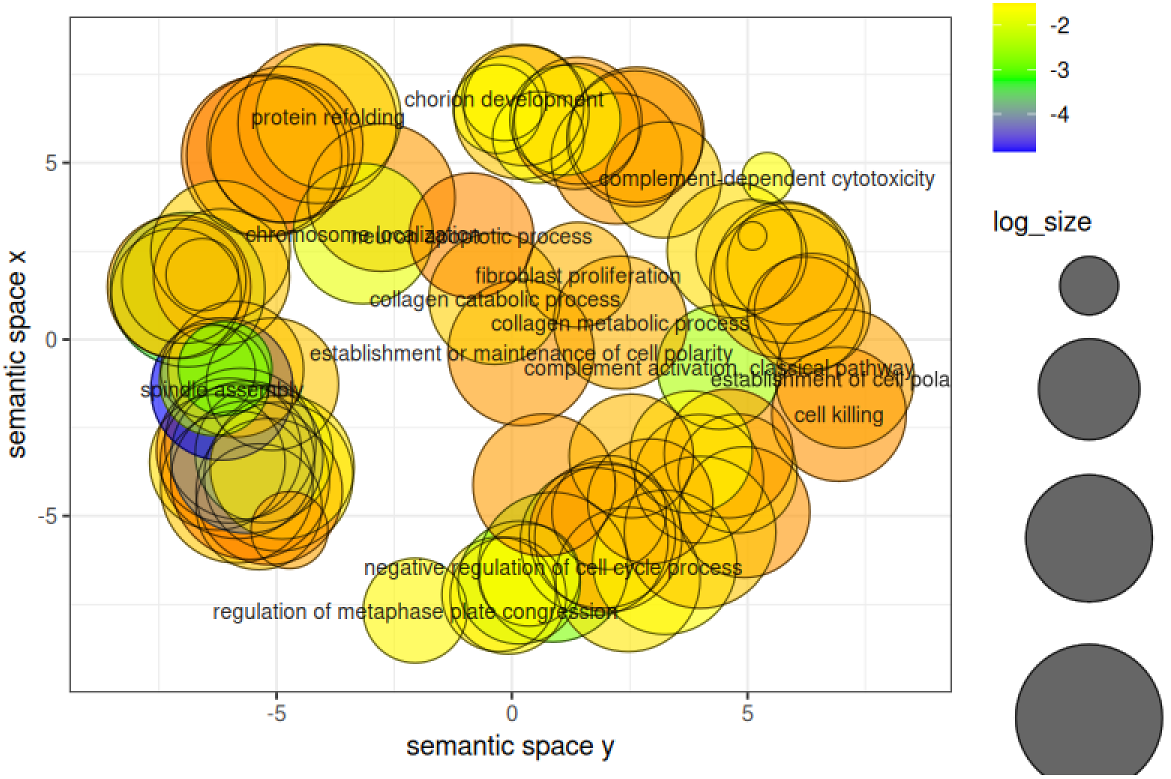

Figure 6: Scatter plot of enriched GO-BP terms among genes over-expressed in COVID-19 patients versus healthy controls, visualized with REVIGO. Bubble size reflects the number of associated genes (log_size) and color represents significance (–log10(adjusted p-value)). Mitotic, immune, and tissue-remodeling processes cluster into distinct semantic regions.

The REVIGO scatter plot places the enriched GO Biological Process terms in the context of their semantic similarity. Each bubble represents one term, whose size (log_size) indicates the number of associated genes, and whose colour encodes the level of significance (–log10(adjusted p-value)). The coordinates on the "semantic space x" and "semantic space y" axes are arbitrary, but they cluster biologically related processes together [5].

In the lower-left quadrant, a cluster of mitotic processes—"spindle assembly", "nuclear division", and "chromosome segregation"—stands out, characterised by small bubbles (few genes) in blue-green tones, reflecting high significance. To the right, a second group clusters immune-defence pathways ("complement activation", "cell killing", "complement-dependent cytotoxicity") with medium-sized bubbles in yellow-orange tones. Finally, in the upper-central region appear tissue-remodelling and developmental processes ("fibroblast proliferation", "collagen catabolic process", "protein refolding", "chorion development") with larger yellow bubbles, indicating moderate enrichment but involving more genes.

Discussion

The integration of RNA-seq counts with GEO metadata and the construction of a `SummarizedExperiment` object allowed us to rigorously normalise and explore the transcriptomic signatures in COVID-19, bacterial pneumonia, and healthy controls. PCA and MDS confirm that cohort is the main source of variation, with a slight batch effect and a secondary impact of sex, which justifies their inclusion in the differential-expression design.

The *voom+limma* pipeline, adjusted for batch and gender, identifies hundreds of genes with $|\log_2 \mathrm{FC}| \geq 1.5$ and $\mathrm{FDR} < 0.05$ in both comparisons. Whereas bacterial pneumonia shows a strong induction of pro-inflammatory genes, COVID-19 is characterised by transcriptional repression and an unexpected activation of mitotic processes in peripheral blood.

GO-BP enrichment and its REVIGO summary reveal three functional axes—mitosis, immune cytotoxicity, and tissue remodelling—suggesting that SARS-CoV-2 affects not only the classical immune response but also the proliferative dynamics of blood cells.

The comparison of our peripheral blood transcriptomic signatures with the findings of McClain et al. (2021) reveals convergence on several biological axes, as well as complementary insights that enrich our interpretation. Similar to that study, we observe marked heterogeneity in interferon responses—with some ISGs up-regulated and others down-regulated—suggesting a characteristic disruption of type I IFN signalling during SARS-CoV-2 infection :contentReferenceoaicite:0.

However, while McClain et al. emphasize an early activation of antiviral defence and B-cell pathways (e.g., CD79A/B, IGH, detectable from day 1) and derive a 23–139-gene signature capable of discriminating COVID-19 from other infections with an auROC aprox. 0.95 :contentReferenceoaicite:1, our analysis unexpectedly highlights a mitotic activation component (spindle assembly, nuclear division, organelle fission) in COVID-19 versus healthy controls.

Moreover, McClain et al. also describe disruptions in coagulation/fibrinolysis pathways (KLKB1, F12, SERPINE1, PROS1) and modulation of IL-1, JAK/STAT, and IL-6 signalling—characterized by a hypo-inflammatory profile early in disease and heterogeneous elevations in later phases :contentReferenceoaicite:2. Although our design did not include influenza or seasonal coronavirus controls, the pronounced induction of mitotic processes we identify may reflect a compensatory mechanism or the proliferation of a specific cell subset (e.g., progenitors or plasmablasts) in response to the inflammatory imbalance and haemostatic alterations noted by McClain et al.

These findings confirm that beyond the classic interferon signature, SARS-CoV-2 induces a multifaceted reconfiguration of the peripheral transcriptome:

- **Antiviral and humoral response**: elevated ISGs and immunoglobulin genes, consistent with McClain et al. :contentReferenceoaicite:3.
- **Inflammatory and haemostatic modulation**: atypical IL-1/JAK-STAT patterns and coagulation/fibrinolysis pathway changes.
- **Mitotic activation**: our most distinctive finding, suggesting peripheral blood cell proliferation potentially linked to repair mechanisms or immune lineage expansion.

Conclusions

- Cohort (COVID-19, bacterial pneumonia, healthy) remains the principal driver of peripheral-blood transcriptomic variation, with batch and sex effects appropriately adjusted in the design.

- Our *voom+limma* analysis ($|\log_2 \mathrm{FC}| > 1.5$, FDR < 0.05) confirms McClain et al.'s interferon- and immunoglobulin-driven signature in COVID-19—yet also reveals a novel mitotic activation programme (spindle assembly, nuclear division, organelle fission) not highlighted previously.

- Bacterial pneumonia displays a robust induction of pro-inflammatory and coagulation/fibrinolysis pathways, whereas COVID-19 samples show both transcriptional repression in classic immune genes and up-regulation of cell-cycle processes, suggesting compensatory proliferation (e.g., plasmablasts or progenitors).

- GO-BP enrichment and REVIGO summarization unify three axes—mitosis, immune cytotoxicity, and tissue remodelling—underscoring that SARS-CoV-2 reprogrammes not only antiviral immunity but also haemostatic balance and cell-proliferation programmes in peripheral blood.

References

[1] McClain, Micah T et al. "Dysregulated transcriptional responses to SARS-CoV-2 in the periphery." Nature communications vol. 12,1 1079. 17 Feb. 2021, doi:10.1038/s41467-021-21289-y

[2] Chen Y, Lun ATL, and Smyth, GK (2016). From reads to genes to pathways: differential expression analysis of RNA-Seq experiments using Rsubread and the edgeR quasi-likelihood pipeline. F1000Research 5, 1438. http://f1000research.com/articles/5-1438

[3] Greenacre, M., Groenen, P.J.F., Hastie, T. et al. Principal component analysis. Nat Rev Methods Primers 2, 100 (2022). https://doi.org/10.1038/s43586-022-00184-w

[4] Nasir Saeed, Haewoon Nam, Mian Imtiaz Ul Haq, and Dost Bhatti Muhammad Saqib. 2018. A Survey on Multidimensional Scaling. ACM Comput. Surv. 51, 3, Article 47 (May 2019), 25 pages. https://doi.org/10.1145/3178155

[5] Supek F, Bošnjak M, Škunca N, Šmuc T (2011) REVIGO Summarizes and Visualizes Long Lists of Gene Ontology Terms. PLOS ONE 6(7): e21800. https://doi.org/10.1371/journal.pone.0021800

Anexus

```r
# A plotting R script produced by the Revigo server at http://revigo.irb.hr/
# If you found Revigo useful in your work, please cite the following reference:
# Supek F et al. "REVIGO summarizes and visualizes long lists of Gene Ontology
# terms" PLoS ONE 2011. doi:10.1371/journal.pone.0021800

# --------------------------------------------------------------------------
# If you don't have the ggplot2 package installed, uncomment the following line:
# install.packages( "ggplot2" );
library( ggplot2 );

# --------------------------------------------------------------------------
# If you don't have the scales package installed, uncomment the following line:
# install.packages( "scales" );
library( scales );

# --------------------------------------------------------------------------
# Here is your data from Revigo. Scroll down for plot configuration options.

revigo.names <- c("term_ID","description","frequency","plot_X","plot_Y","log_size","value","uniqueness"
revigo.data <- rbind(c("GO:0000002","mitochondrial genome maintenance",0.06554336753744247,-6.259821644
c("GO:0000280","nuclear division",0.43374453844241945,-5.770988064614843,-2.9554586835027727,5.24606259
c("GO:0001788","antibody-dependent cellular cytotoxicity",0.0001501425295649101,5.1039142827992015,2.93
c("GO:0001890","placenta development",0.012919641601413328,0.25489092657336215,6.6209013916071555,3.720
c("GO:0001906","cell killing",0.05707631341115902,6.941402586438603,-2.1155370837861773,4.365300748637
c("GO:0002040","sprouting angiogenesis",0.025684217967374374,1.4029727926060354,6.19137302452011,4.0185
c("GO:0002437","inflammatory response to antigenic stimulus",0.005311599652476655,5.080435132688191,2.4
c("GO:0002520","immune system development",0.03717135215556184,2.21985129437037,5.128085398719903,4.179
c("GO:0002526","acute inflammatory response",0.014610591073726333,5.854936530155104,2.147492382388848,3
c("GO:0006260","DNA replication",1.4421509940592292,-5.31921181180662,5.192584920007242,5.767837581402
c("GO:0006278","RNA-templated DNA biosynthetic process",0.17927264165344634,-4.880888328760733,5.361997
c("GO:0006310","DNA recombination",1.827734239452524,-5.190752862775078,5.210824089798946,5.870739739
c("GO:0006457","protein folding",1.1925033490398829,-4.132689541310154,6.113551762643961,5.685286610314
c("GO:0006958","complement activation, classical pathway",0.019520990196382,4.399119947275016,-0.783434
c("GO:0006959","humoral immune response",0.05994378959055443,4.721057436911085,2.5211645126302717,4.386
c("GO:0007004","telomere maintenance via telomerase",0.0454661115757872,-5.089576192520205,-1.266560507
c("GO:0007005","mitochondrion organization",0.6594013763196482,-5.830468901922794,-3.4586017913808442,5
c("GO:0007059","chromosome segregation",0.6817307702247771,-6.891164964139785,1.4156834848039654,5.4424
c("GO:0007163","establishment or maintenance of cell polarity",0.22665860852530942,0.194714954497797,-0
c("GO:0007605","sensory perception of sound",0.0593038378251958,2.6431752443178698,5.80013234361473,4.38
c("GO:0009408","response to heat",0.3335994712226244,5.787098437370566,1.571858722571108,5.132054664397
c("GO:0010458","exit from mitosis",0.0015604977662975902,-6.458901012246711,-0.89027388044292,2.8027737
c("GO:0010639","negative regulation of organelle organization",0.26324415868027373,1.74383376037818,-5.
c("GO:0010948","negative regulation of cell cycle process",0.3342984954586315,0.8714722922661182,-6.447
c("GO:0022411","cellular component disassembly",0.49554664950545757,-5.846147052397789,-4.1797490362588
c("GO:0030010","establishment of cell polarity",0.09436581050801783,7.066748592738407,-1.10862276497943
c("GO:0030199","collagen fibril organization",0.023070261141178723,-5.374682760114819,-4.74136768034249
c("GO:0030574","collagen catabolic process",0.04133842268922402,-0.36580287693507824,1.158768377845892
c("GO:0031503","protein-containing complex localization",0.2665596010954202,-2.773424267002288,4.021334
c("GO:0032147","activation of protein kinase activity",0.0308678272667793,3.9683956675947796,-3.9500103
c("GO:0032200","telomere organization",0.19561848655247202,-6.013691890372712,-3.150034038569182,4.9002
c("GO:0032211","negative regulation of telomere maintenance via telomerase",0.007169921124960378,2.0613
c("GO:0032271","regulation of protein polymerization",0.22037477446007636,1.918005063537686,-5.5381403
c("GO:0032886","regulation of microtubule-based process",0.17552892382609833,2.461564857015948,-6.79856
c("GO:0032963","collagen metabolic process",0.04852065057890283,2.298348586547254,0.47956362460884305,4
c("GO:0032984","protein-containing complex disassembly",0.3227941317998383,-4.957254803089621,-3.890216
c("GO:0042026","protein refolding",0.19468809513975832,-3.8988505751852345,6.30482629217927,4.89817099
c("GO:0042742","defense response to bacterium",0.1321229646641772,5.799335990256828,1.9033505694017834,
c("GO:0043029","T cell homeostasis",0.008279991302563239,3.2186282284982073,4.5221390721068655,3.526985
c("GO:0043062","extracellular structure organization",0.23198743764839191,-5.15656387046929,-4.3112189
c("GO:0043244","regulation of protein-containing complex disassembly",0.24929813290134295,2.0458276516
c("GO:0043523","regulation of neuron apoptotic process",0.03835034021558793,4.932192615466645,-4.896813
c("GO:0043525","positive regulation of neuron apoptotic process",0.01144036848225741,4.78767044911135,-
c("GO:0044771","meiotic cell cycle phase transition",0.0004208913533704856,-6.579494231783464,1.837671
c("GO:0044786","cell cycle DNA replication",0.09954695845447907,-6.191460291320214,2.518509336677662,4
c("GO:0044839","cell cycle G2/M phase transition",0.01907548531357464,-6.969446166277071,1.648454514730
c("GO:0045229","external encapsulating structure organization",1.2096737471750436,-5.562604688414026,-3
c("GO:0045766","positive regulation of angiogenesis",0.03182037085598619,4.597736423635044,-3.233725421
c("GO:0046785","microtubule polymerization",0.08265961590784222,-5.266901704474342,-3.242334611543995,4
c("GO:0048144","fibroblast proliferation",0.003384360297569695,1.4031204053908497,1.835179295126816,3.
c("GO:0048285","organelle fission",0.5008139448049479,-5.629838905453105,-3.335624603230398,5.308504653
c("GO:0048732","gland development",0.05944413494298596,0.20536382638593084,6.446338919971726,4.38295310
c("GO:0050000","chromosome localization",0.10568311134309089,-3.1617716107234943,2.9967115766112338,4.6
c("GO:0050954","sensory perception of mechanical stimulus",0.06226238406350763,2.604153387429476,5.6928
c("GO:0051225","spindle assembly",0.14606652909016435,-6.159497641503413,-1.4008181990276294,4.7733841
c("GO:0051258","protein polymerization",0.1736164525888535,-4.881714788680789,-3.617368096996544,4.8484
c("GO:0051293","establishment of spindle localization",0.06700541118599257,-5.868616255116801,-0.844038
c("GO:0051302","regulation of cell division",0.16835161864230688,3.2425893410916666,-6.312080557266651,
c("GO:0051304","chromosome separation",0.03391498221106387,-7.132317713272933,1.316494713358019,4.13924
c("GO:0051306","mitotic sister chromatid separation",0.005225452299447608,-6.3142965817512104,-1.164306
c("GO:0051338","regulation of transferase activity",0.45600747581806883,3.9902690783390526,-5.45937091
c("GO:0051402","neuron apoptotic process",0.019794200373131263,-0.8565707583872858,2.9425696930900758,3
c("GO:0051653","spindle localization",0.06821147412839923,-6.169140685732736,1.780014616353336,4.442699
c("GO:0051988","regulation of attachment of spindle microtubules to kinetochore",0.007211764125003059,-
c("GO:0060707","trophoblast giant cell differentiation",0.0012355991777308994,0.5629729151067154,5.7203
c("GO:0060717","chorion development",0.0007974783537546043,-0.16160346920657387,6.821490010453913,2.511
c("GO:0061008","hepaticobiliary system development",0.022952116199881745,1.356530068918793,6.0293686499
c("GO:0061640","cytoskeleton-dependent cytokinesis",0.16478019552101697,-7.073486059850885,1.4914617355
c("GO:0070365","hepatocyte differentiation",0.005104846005206943,1.1277737717701781,6.20616376053117,3.
c("GO:0071897","DNA biosynthetic process",0.7699653505500705,-4.820524587414945,5.526678812668085,5.495
c("GO:0090235","regulation of metaphase plate congression",0.0031283795914262415,-2.053249054159235,-7.
c("GO:0090306","meiotic spindle assembly",0.0014521982367753598,-6.092643768628381,-0.8206609700366004,
c("GO:0097278","complement-dependent cytotoxicity",0.00019444688255127702,5.4181319425278796,4.5728374
c("GO:0140588","chromatin looping",0.0005267295299490289,-4.726720543454419,-5.403390483432511,2.332438
c("GO:1901976","regulation of cell cycle checkpoint",0.016185856957686044,0.15482612818437633,-6.884488
c("GO:1902412","regulation of mitotic cytokinesis",0.008787030008962771,-0.07702572713827012,-7.2580428
c("GO:1902423","regulation of attachment of mitotic spindle microtubules to kinetochore",0.002584420590
c("GO:1903867","extraembryonic membrane development",0.0010313068834048743,-0.30427834620046407,6.52202
c("GO:1990823","response to leukemia inhibitory factor",0.008910097656147123,6.2327414655180196,0.69715
c("GO:1990830","cellular response to leukemia inhibitory factor",0.008686114538271603,6.375027234132312
c("GO:2000105","positive regulation of DNA-templated DNA replication",0.015836344839682485,3.801000995
c("GO:2000241","regulation of reproductive process",0.14026758155483765,0.6838688066694735,-4.126408823
c("GO:2000278","regulation of DNA biosynthetic process",0.018755509430895326,2.5332710630269815,-3.3052
c("GO:2001252","positive regulation of chromosome organization",0.020535067609181064,2.936905389392559,

one.data <- data.frame(revigo.data);
names(one.data) <- revigo.names;
one.data <- one.data [(one.data$plot_X != "null" & one.data$plot_Y != "null"), ];
one.data$plot_X <- as.numeric( as.character(one.data$plot_X) );
one.data$plot_Y <- as.numeric( as.character(one.data$plot_Y) );
one.data$log_size <- as.numeric( as.character(one.data$log_size) );
one.data$value <- as.numeric( as.character(one.data$value) );
one.data$frequency <- as.numeric( as.character(one.data$frequency) );
one.data$uniqueness <- as.numeric( as.character(one.data$uniqueness) );
one.data$dispensability <- as.numeric( as.character(one.data$dispensability) );
#head(one.data);


# --------------------------------------------------------------------------
# Names of the axes, sizes of the numbers and letters, names of the columns,
# etc. can be changed below

p1 <- ggplot( data = one.data );
p1 <- p1 + geom_point( aes( plot_X, plot_Y, colour = value, size = log_size), alpha = I(0.6) );
p1 <- p1 + scale_colour_gradientn( colours = c("blue", "green", "yellow", "red"), limits = c( min(one.da
p1 <- p1 + geom_point( aes(plot_X, plot_Y, size = log_size), shape = 21, fill = "transparent", colour =
p1 <- p1 + scale_size( range=c(5, 30)) + theme_bw(); # + scale_fill_gradientn(colours = heat_hcl(7), li
ex <- one.data [ one.data$dispensability < 0.15, ];
p1 <- p1 + geom_text( data = ex, aes(plot_X, plot_Y, label = description), colour = I(alpha("black", 0.
p1 <- p1 + labs (y = "semantic space x", x = "semantic space y");
p1 <- p1 + theme(legend.key = element_blank()) ;
one.x_range = max(one.data$plot_X) - min(one.data$plot_X);
one.y_range = max(one.data$plot_Y) - min(one.data$plot_Y);
p1 <- p1 + xlim(min(one.data$plot_X)-one.x_range/10,max(one.data$plot_X)+one.x_range/10);
p1 <- p1 + ylim(min(one.data$plot_Y)-one.y_range/10,max(one.data$plot_Y)+one.y_range/10);


# --------------------------------------------------------------------------
# Output the plot to screen

p1;
```