Step 1.1: Pre-processing without volunteers 9 & 10

Carlos Gallardo & Christian Oertlin

17 April, 2023

```
# Import libraries and helper functions
source("code/helper_functions.R")
library(tidyverse)
library(magrittr)
library(patchwork)
library(DESeq2)
library(limma)
library(variancePartition)
library(RColorBrewer)

# Colors
colPals <- vector(mode = "list")
colPals$time <- setNames(c("#FBAA3E", "#2C83BE", "#3EB6BD", "#A3D5B3", "#CD71A8"),
                         nm = c("day0", "day7", "day14", "day21", "day28"))
colPals$time_light <- setNames(c("#FDD6A1", "#A2CDE9", "#AFE2E5", "#DDF0E3", "#E8BDD6"),
                               nm = c("day0", "day7", "day14", "day21", "day28"))
colPals$time_dark <- setNames(c("#D87E04", "#174564", "#1F5C60", "#49A065", "#AA3C7E"),
                              nm = c("day0", "day7", "day14", "day21", "day28"))
colPals$inferno <- c("#000004", "#420A68", "#932667", "#DD513A", "#FCA50A", "#FCFFA4")
```

Load data

Background annotation

```
ann_data <- read.table(
  file = 'data/resources/gene_annotation_ensembl_v104.txt',
  stringsAsFactors = FALSE,
  sep = "\t",
  header = TRUE,
  fill = FALSE,
  quote = "") %>%
  dplyr::rename(Geneid = ensembl_gene_id)
```

RNAseq expression data

> From a previous pre-processing we saw that volunteers 9 and 10 have opposite fold change profiles when compared to other volunteers. This includes genes that are highly expressed in blood monocytes (e.g., CDKN1A, NR4A1, NR4A2, MYADM, IRS2 and CD83). Here we perform a pre-processing where these volunteers have been removed according to our exclusion criteria.

```
RNAseq <- vector("list")

RNAseq[["unfilt"]][["rawdata"]] <- read.table(file = 'data/rnaseq/rnaseq_count_mtx.tsv',
                                              stringsAsFactors = FALSE,
                                              sep = "\t",
                                              header = TRUE)

# Make duplicated gene names unique
RNAseq[["unfilt"]][["rawdata"]] <- RNAseq[["unfilt"]][["rawdata"]] %>%
  mutate(GeneSymbol = uniquify(plyr::mapvalues(.$Geneid,
                                               from = ann_data$Geneid,
                                               to = ann_data$external_gene_name,
                                               warn_missing = F), sep = '_'))

RNAseq[["unfilt"]][["annotation"]] <- RNAseq[["unfilt"]][["rawdata"]] %>%
  select(Geneid, GeneSymbol) %>%
  inner_join(ann_data, by = "Geneid") %>%
  select(-c(external_gene_name))

RNAseq[["unfilt"]][["design"]] <- read.table(file = 'data/RNAseq/rnaseq_design_mtx.tsv',
                                             stringsAsFactors = FALSE,
                                             sep = "\t",
                                             header = TRUE) %>%
  mutate(sample = factor(sample, levels = sample),
         batch = factor(batch, levels = unique(batch)),
         volunteer = factor(volunteer, levels = unique(volunteer)),
         time = factor(time, levels = unique(time)))

RNAseq[["unfilt"]][["counts"]] <- RNAseq[["unfilt"]][["rawdata"]] %>%
  select(-c(1:3)) %>%
  column_to_rownames("GeneSymbol")
```

Pre-processing

Filtering zero count genes

```
paste("Raw feature count:", nrow(RNAseq$unfilt$counts))
```

```
## [1] "Raw feature count: 60649"
```

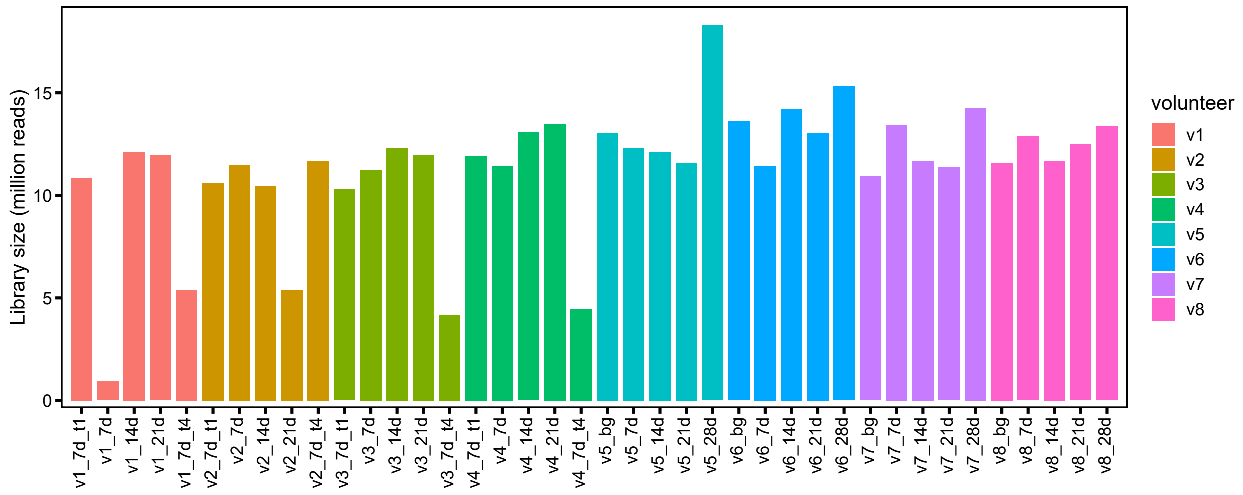
```
tokeep <- rowSums(RNAseq$unfilt$counts) > 0
paste("Non-zero feature count:", sum(tokeep))
```

```
## [1] "Non-zero feature count: 42112"
```

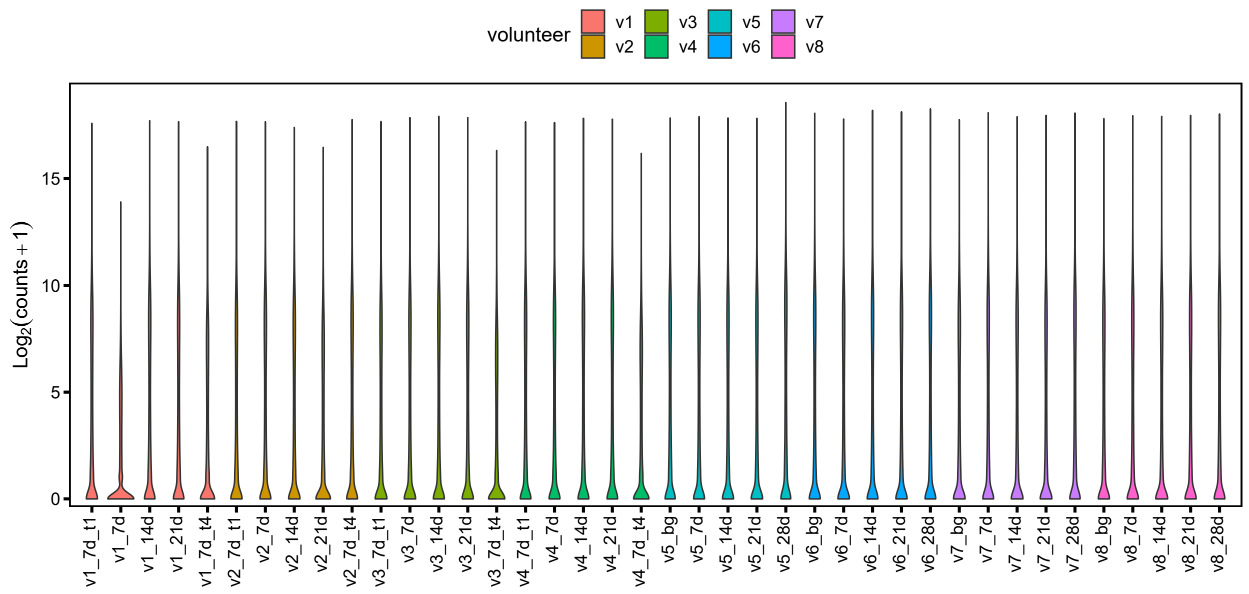
```
RNAseq$unfilt$rawdata <- RNAseq$unfilt$rawdata[tokeep,]
RNAseq$unfilt$annotation <- RNAseq$unfilt$annotation[tokeep,]
RNAseq$unfilt$counts <- RNAseq$unfilt$counts[tokeep,]
rm(tokeep)
```

Library sizes and gene expression distributions

```
df <- RNAseq$unfilt$design
df$lib.size <- colSums(RNAseq$unfilt$counts)

ggplot(df, aes(x=sample, y=lib.size/1e6, fill=volunteer)) +
  geom_bar(stat = "identity", width = 0.8) +
  xlab("") +
  ylab("Library size (million reads)") +
  scale_x_discrete(expand =expansion(mult = c(.02, .02))) +
  scale_y_continuous(expand =expansion(mult = c(.02, .05))) +
  theme_custom(
    axis.text.x.bottom = element_text(angle = 90, hjust = 1, vjust = 0.3)
  )
```

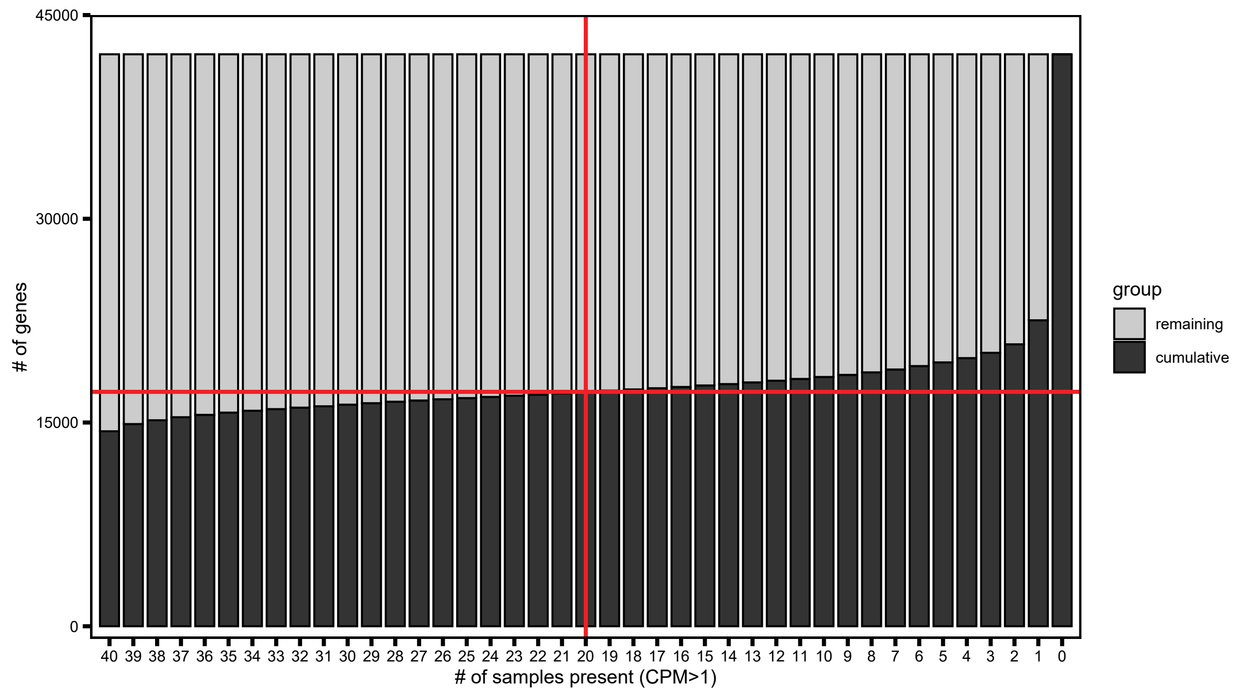


```
df <- RNAseq$unfilt$counts %>%
  rownames_to_column(var = "geneID") %>%
  pivot_longer(cols = c(2:length(.)),
               names_to = "sample") %>%
  dplyr::rename(counts = value)

df$volunteer <- rep(RNAseq$unfilt$design$volunteer, dim(RNAseq$unfilt$counts)[1])
df$time <- rep(RNAseq$unfilt$design$time, dim(RNAseq$unfilt$counts)[1])
df$sample <- factor(df$sample, levels = names(RNAseq$unfilt$counts))

ggplot(df, aes(x=sample, y=log2(counts+1), fill=volunteer)) +
  geom_violin(scale = "area") +
  xlab("") +
  ylab(expression(Log[2](counts+1))) +
  scale_x_discrete(expand =expansion(mult = c(.02, .02))) +
  scale_y_continuous(expand =expansion(mult = c(.02, .05))) +
  theme_custom(
    axis.text.x.bottom = element_text(angle = 90, hjust = 1, vjust = 0.3),
    legend.position = "top"
  )
```



Filtering low abundance genes

```
# Calculate CPM
RNAseq[["unfilt"]][["cpm"]] <- cpm.normalize(RNAseq$unfilt$counts)

abovethresh <- RNAseq$unfilt$cpm > 1

df <- data.frame(samples = factor(seq(0,ncol(RNAseq$unfilt$cpm),1),
                                  levels = rev(seq(0,ncol(RNAseq$unfilt$cpm),1))),
                 genes = c(table(rowSums(abovethresh)))) %>%
  mutate(cumulative = rev(cumsum(rev(genes)))) %>%
  mutate(remaining = sum(genes)-cumulative) %>%
  pivot_longer(cols = c("cumulative","remaining"),
               names_to = "group") %>%
  mutate(group = factor(group, levels = c("remaining", "cumulative")))

ggplot(data=df, aes(x=samples, y=value, fill=group)) +
  geom_bar(color="black", size=0.5, width=0.8, position="stack", stat="identity") +
  geom_hline(yintercept = unlist(df[df$samples == "20" & df$group == "cumulative", "value"]),
             linetype="solid", size=1, color="#EF2126") +
  geom_vline(xintercept = "20", linetype="solid", size=1, color="#EF2126") +
  xlab("# of samples present (CPM>1)") +
  ylab("# of genes") +
  scale_x_discrete(expand =expansion(mult = c(.02, .02))) +
  scale_y_continuous(expand =expansion(mult = c(.02, .00)),
                     limits = c(0,45000), breaks = seq(0,45000,15000)) +
  scale_fill_manual(values = c("grey80", "grey20")) +
  theme_custom(base_size = 8)
```



```
tokeep <- rowSums(abovethresh) >= 20
paste("Pre-filtering gene count:", length(tokeep))
```

```
## [1] "Pre-filtering gene count: 42112"
```

```
paste("Genes below abundance threshold:", length(tokeep)-sum(tokeep))
```

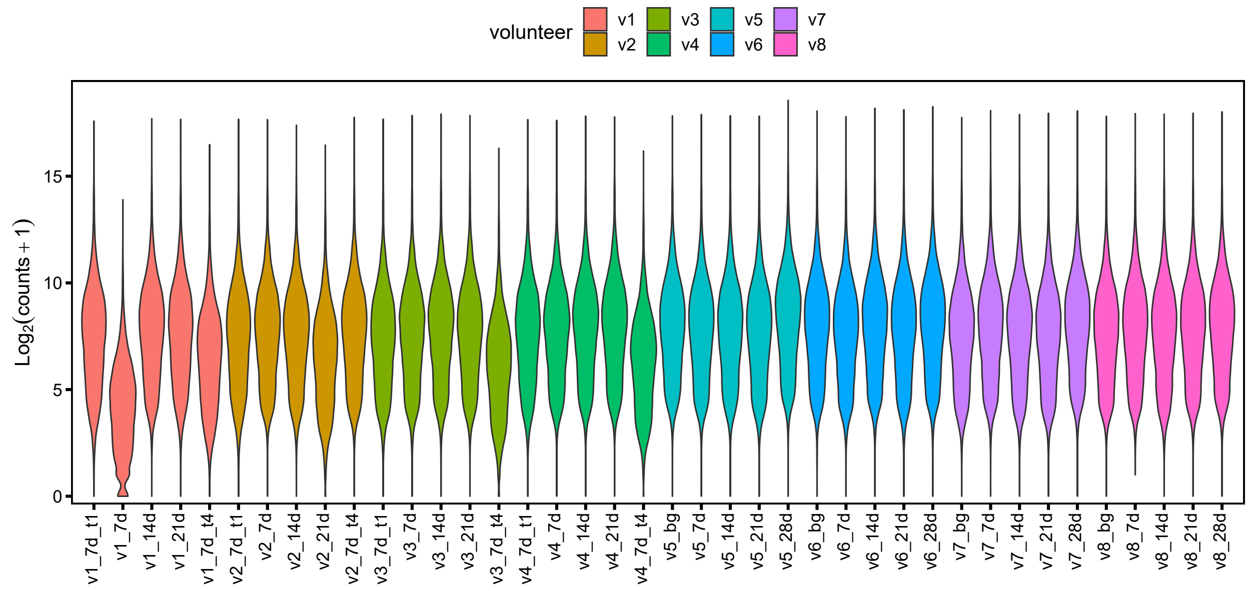
```
## [1] "Genes below abundance threshold: 24853"
```

```
paste("Remaining genes:", sum(tokeep))
```

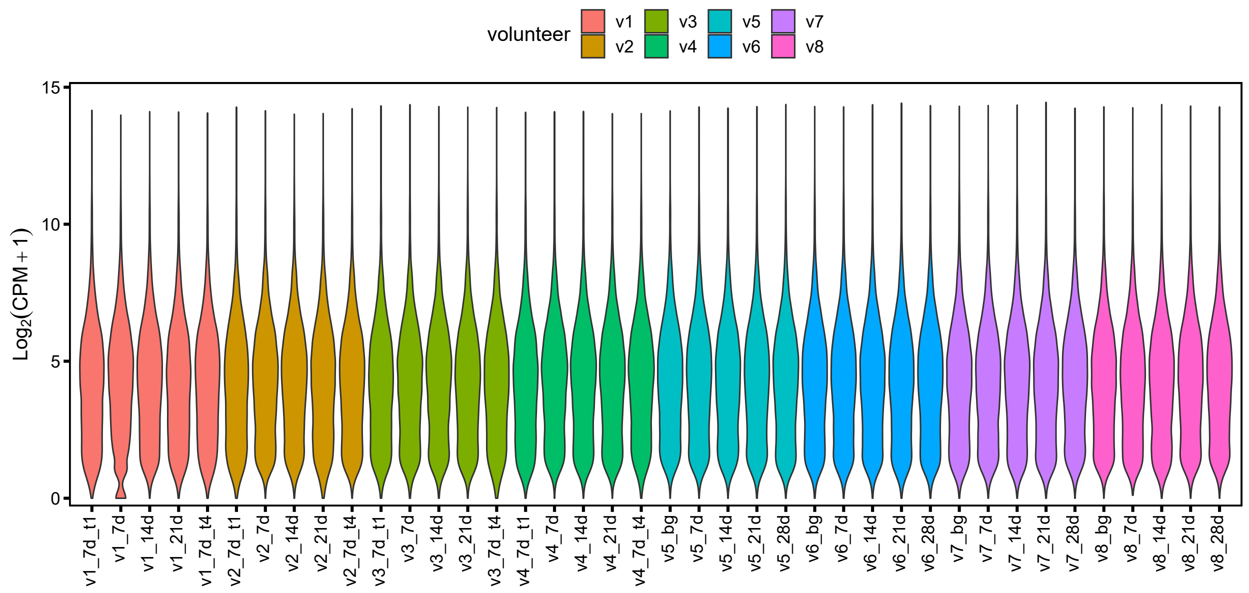
```
## [1] "Remaining genes: 17259"
```

```
# Filter genes
RNAseq[["filt"]][["rawdata"]] <- RNAseq$unfilt$rawdata[tokeep,]
RNAseq[["filt"]][["annotation"]] <- RNAseq$unfilt$annotation[tokeep,]
RNAseq[["filt"]][["design"]] <- RNAseq$unfilt$design
RNAseq[["filt"]][["counts"]] <- RNAseq$unfilt$counts[tokeep,]

# Normalize
RNAseq[["filt"]][["cpm"]] <- cpm.normalize(RNAseq$filt$counts)
```

```
# Expression distribution post-filtering
df <- RNAseq$filt$counts %>%
  rownames_to_column(var = "geneID") %>%
  pivot_longer(cols = c(2:length(.)),
               names_to = "sample") %>%
  dplyr::rename(counts = value)

df$volunteer <- rep(RNAseq$filt$design$volunteer, dim(RNAseq$filt$counts)[1])
df$time <- rep(RNAseq$filt$design$time, dim(RNAseq$filt$counts)[1])
df$sample <- factor(df$sample, levels = names(RNAseq$filt$counts))

ggplot(df, aes(x=sample, y=log2(counts+1), fill=volunteer)) +
  geom_violin(scale = "area") +
  xlab("") +
  ylab(expression(Log[2](counts+1))) +
  scale_x_discrete(expand =expansion(mult = c(.02, .02))) +
  scale_y_continuous(expand =expansion(mult = c(.02, .05))) +
  theme_custom(
    axis.text.x.bottom = element_text(angle = 90, hjust = 1, vjust = 0.3),
    legend.position = "top"
  )
```



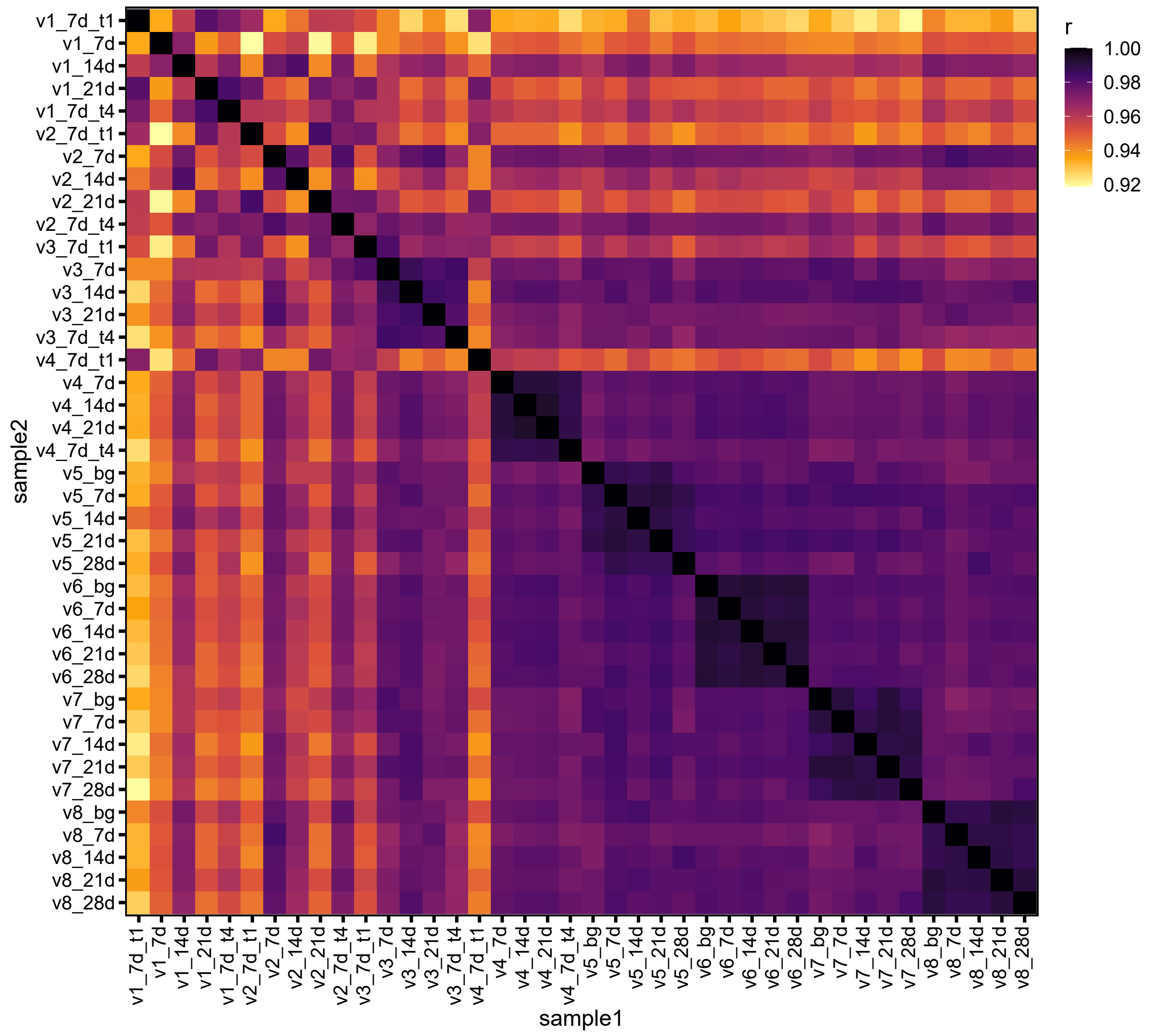
```r
# Normalized expression distribution post-filtering
df <- RNAseq$filt$counts %>%
  rownames_to_column(var = "geneID") %>%
  pivot_longer(cols = c(2:length(.)),
               names_to = "sample") %>%
  dplyr::rename(counts = value)

df$volunteer <- rep(RNAseq$filt$design$volunteer, dim(RNAseq$filt$counts)[1])
df$time <- rep(RNAseq$filt$design$time, dim(RNAseq$filt$counts)[1])
df$sample <- factor(df$sample, levels = names(RNAseq$filt$counts))

df$cpm <- RNAseq$filt$cpm %>%
  pivot_longer(cols = c(1:length(.)),
               names_to = "sample") %>%
  select(value) %>%
  unlist()

ggplot(df, aes(x=sample, y=log2(cpm+1), fill=volunteer)) +
  geom_violin(scale = "area") +
  xlab("") +
  ylab(expression(Log[2](CPM+1))) +
  scale_x_discrete(expand =expansion(mult = c(.02, .02))) +
  scale_y_continuous(expand =expansion(mult = c(.02, .05))) +
  theme_custom(
    axis.text.x.bottom = element_text(angle = 90, hjust = 1, vjust = 0.3),
    legend.position = "top"
  )
```



Transcriptome differences

Sample correlations

```
df <- cor(log2(RNAseq$filt$cpm+1), method = "spearman") %>%
  as.data.frame() %>%
  rownames_to_column(var = "sample1") %>%
  mutate(across(everything(), as.character)) %>%
  pivot_longer(cols = c(2:length(.)),
               names_to = "sample2") %>%
  dplyr::rename(r = value) %>%
  mutate(sample1 = factor(sample1, levels = names(RNAseq$filt$counts)),
         sample2 = factor(sample2, levels = names(RNAseq$filt$counts)),
         r = as.numeric(r))

ggplot(df, aes(x=sample1, y=sample2, fill= r)) +
  geom_tile() +
  scale_y_discrete(limits=rev) +
  scale_fill_gradientn(colours = rev(colPals$inferno)) +
  theme_custom(
    axis.text.x.bottom = element_text(angle = 90, hjust = 1, vjust = 0.3),
    legend.position = "right",
    legend.justification = "top"
  )
```



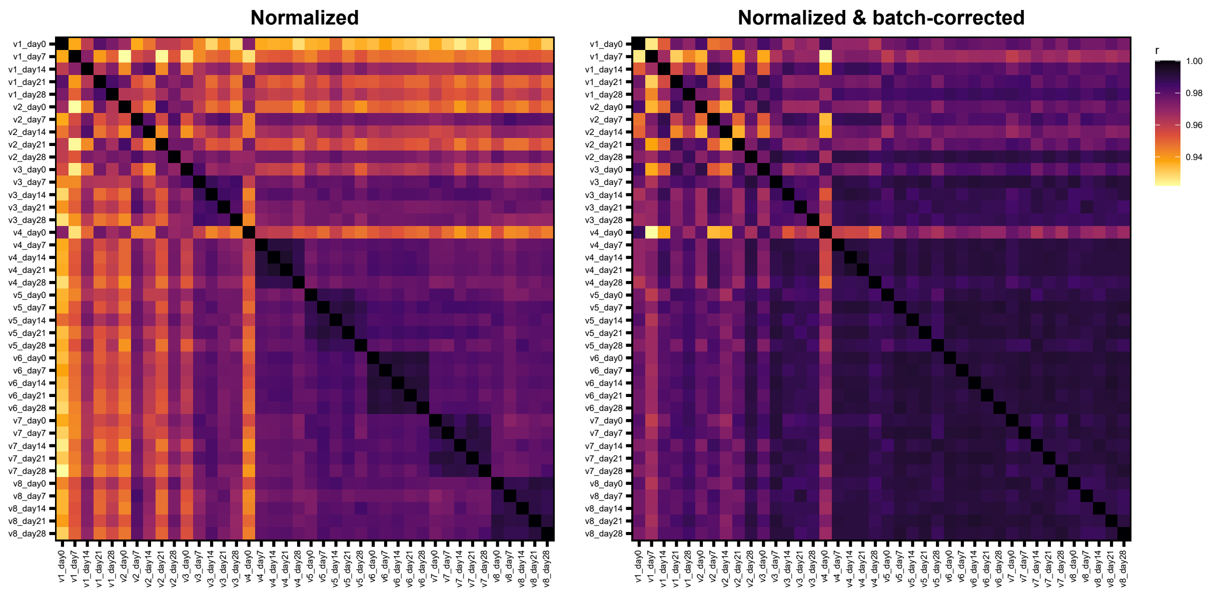
DESeq2 analysis

```
# DESeq2 pipeline
dsData <- DESeqDataSetFromMatrix(countData = RNAseq$filt$counts,
                                 colData = RNAseq$filt$design,
                                 design = ~volunteer + time)
dsData <- estimateSizeFactors(dsData)
dsData <- DESeq(dsData, test = "LRT", reduced = ~volunteer)

RNAseq$filt[["DESeq_norm"]] <- counts(dsData, normalized=TRUE) %>% as.data.frame()
RNAseq$filt[["DESeq_vst"]] <- assay(vst(dsData, blind=FALSE)) %>% as.data.frame()
RNAseq$filt[["DESeq_rlog"]] <- assay(rlog(dsData, blind=FALSE)) %>% as.data.frame()

DESeq2_DEGs <- list(
  GLMtime = results(dsData, test = "LRT", independentFiltering = T),
  day7Vsday0 = results(dsData, contrast=c("time", "day7", "day0"), test = "Wald", independentFiltering = T),
  day14Vsday0 = results(dsData, contrast=c("time", "day14", "day0"), test = "Wald", independentFiltering = T),
  day21Vsday0 = results(dsData, contrast=c("time", "day21", "day0"), test = "Wald", independentFiltering = T),
  day28Vsday0 = results(dsData, contrast=c("time", "day28", "day0"), test = "Wald", independentFiltering = T)
)
DESeq2_DEGs <- lapply(DESeq2_DEGs, as.data.frame)
DESeq2_DEGs$GLMtime <- DESeq2_DEGs$GLMtime %>%
  add_column(logFC_day7Vsday0 = DESeq2_DEGs$day7Vsday0$log2FoldChange, .before = "log2FoldChange") %>%
  add_column(logFC_day14Vsday0 = DESeq2_DEGs$day14Vsday0$log2FoldChange, .before = "log2FoldChange") %>%
  add_column(logFC_day21Vsday0 = DESeq2_DEGs$day21Vsday0$log2FoldChange, .before = "log2FoldChange") %>%
```

```r
  add_column(logFC_day28Vsday0 = DESeq2_DEGs$day28Vsday0$log2FoldChange, .before = "log2FoldChange") %>%
  select(-log2FoldChange)
DESeq2_DEGs <- lapply(DESeq2_DEGs, function(x) mutate(x, padj=ifelse(is.na(padj), 1, padj)))
DESeq2_DEGs <- lapply(DESeq2_DEGs, function(x) arrange(x, padj))
DESeq2_DEGs <- lapply(DESeq2_DEGs, rownames_to_column, var = "GeneSymbol")
DESeq2_DEGs <- lapply(DESeq2_DEGs, inner_join, y = RNAseq$filt$annotation, by = "GeneSymbol")

DESeq2_DEGs_filt <- lapply(DESeq2_DEGs, function(x) x %>% filter(padj<0.01))
```

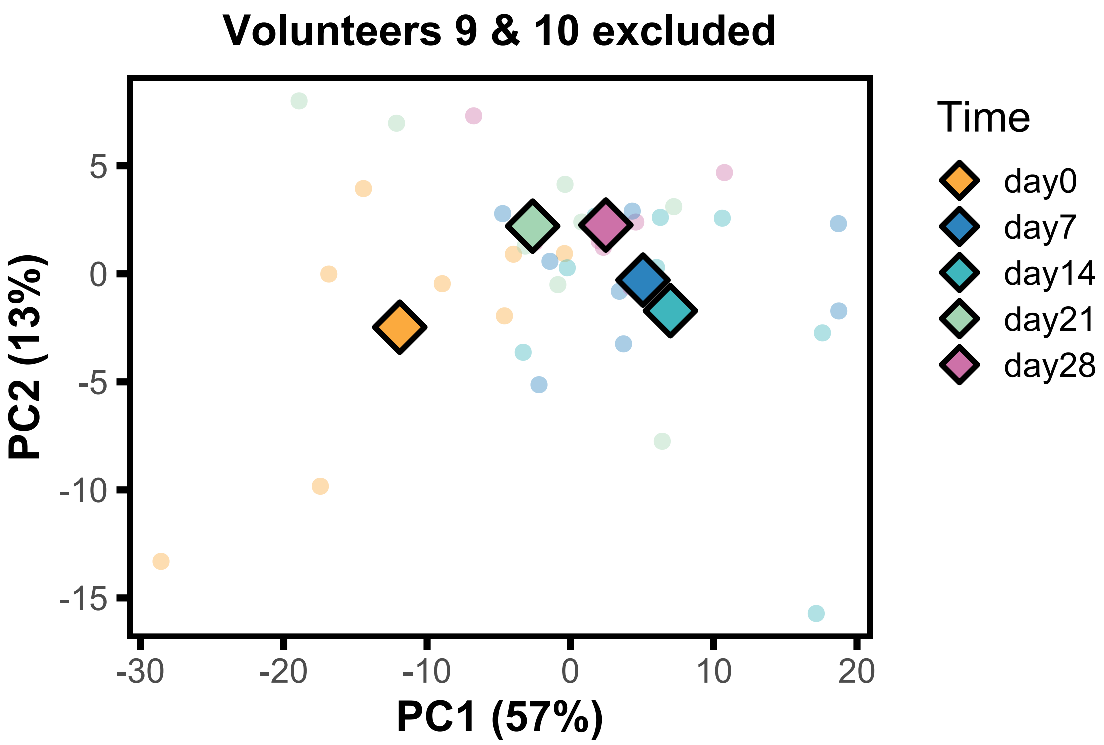
Batch correction

```r
# Remove batch effects from data
design_mtx <- model.matrix(~time, data = RNAseq$filt$design)
RNAseq[["filt"]][["DESeq_vst_nobatch"]] <- removeBatchEffect(
  RNAseq$filt$DESeq_vst,
  batch = dsData$volunteer,
  design = design_mtx) %>% as.data.frame()
RNAseq[["filt"]][["DESeq_rlog_nobatch"]] <- removeBatchEffect(
  RNAseq$filt$DESeq_rlog,
  batch = dsData$volunteer,
  design = design_mtx) %>% as.data.frame()

# Plot sample correlations
df <- cor(RNAseq$filt$DESeq_vst, method = "spearman") %>%
  as.data.frame() %>%
  rownames_to_column(var = "sample1") %>%
  mutate(across(everything(), as.character)) %>%
  pivot_longer(cols = c(2:length(.)),
               names_to = "sample2") %>%
  dplyr::rename(r = value) %>%
  mutate(sample1 = factor(sample1, levels = names(RNAseq$filt$counts)),
         sample2 = factor(sample2, levels = names(RNAseq$filt$counts)),
         r = as.numeric(r))

p1 <- ggplot(df, aes(x=sample1, y=sample2, fill= r)) +
  geom_tile() +
  scale_x_discrete(labels=paste(RNAseq$filt$design$volunteer,
                                RNAseq$filt$design$time,
                                sep = '_')) +
  scale_y_discrete(limits=rev, labels=rev(paste(RNAseq$filt$design$volunteer,
                                                RNAseq$filt$design$time,
                                                sep = '_'))) +
  scale_fill_gradientn(colours = rev(colPals$inferno)) +
  xlab('') +
  ylab('') +
  ggtitle('Normalized') +
  theme_custom(
    base_size = 6,
    axis.text.x.bottom = element_text(angle = 90, hjust = 1, vjust = 0.3),
    legend.position = "none",
    plot.title = element_text(size=14, face='bold', hjust=0.5)
  )

df2 <- cor(RNAseq$filt$DESeq_vst_nobatch, method = "spearman") %>%
  as.data.frame() %>%
  rownames_to_column(var = "sample1") %>%
  mutate(across(everything(), as.character)) %>%
  pivot_longer(cols = c(2:length(.)),
               names_to = "sample2") %>%
  dplyr::rename(r = value) %>%
  mutate(sample1 = factor(sample1, levels = names(RNAseq$filt$counts)),
         sample2 = factor(sample2, levels = names(RNAseq$filt$counts)),
         r = as.numeric(r))

p2 <- ggplot(df2, aes(x=sample1, y=sample2, fill= r)) +
  geom_tile() +
  scale_x_discrete(labels=paste(RNAseq$filt$design$volunteer,
                                RNAseq$filt$design$time,
                                sep = '_')) +
  scale_y_discrete(limits=rev, labels=rev(paste(RNAseq$filt$design$volunteer,
                                                RNAseq$filt$design$time,
                                                sep = '_'))) +
  scale_fill_gradientn(colours = rev(colPals$inferno)) +
  xlab('') +
  ylab('') +
  ggtitle('Normalized & batch-corrected') +
  theme_custom(
    base_size = 6,
    axis.text.x.bottom = element_text(angle = 90, hjust = 1, vjust = 0.3),
    legend.position = "right",
    legend.justification = "top",
    plot.title = element_text(size=14, face='bold', hjust=0.5)
  )

p1 + p2
```



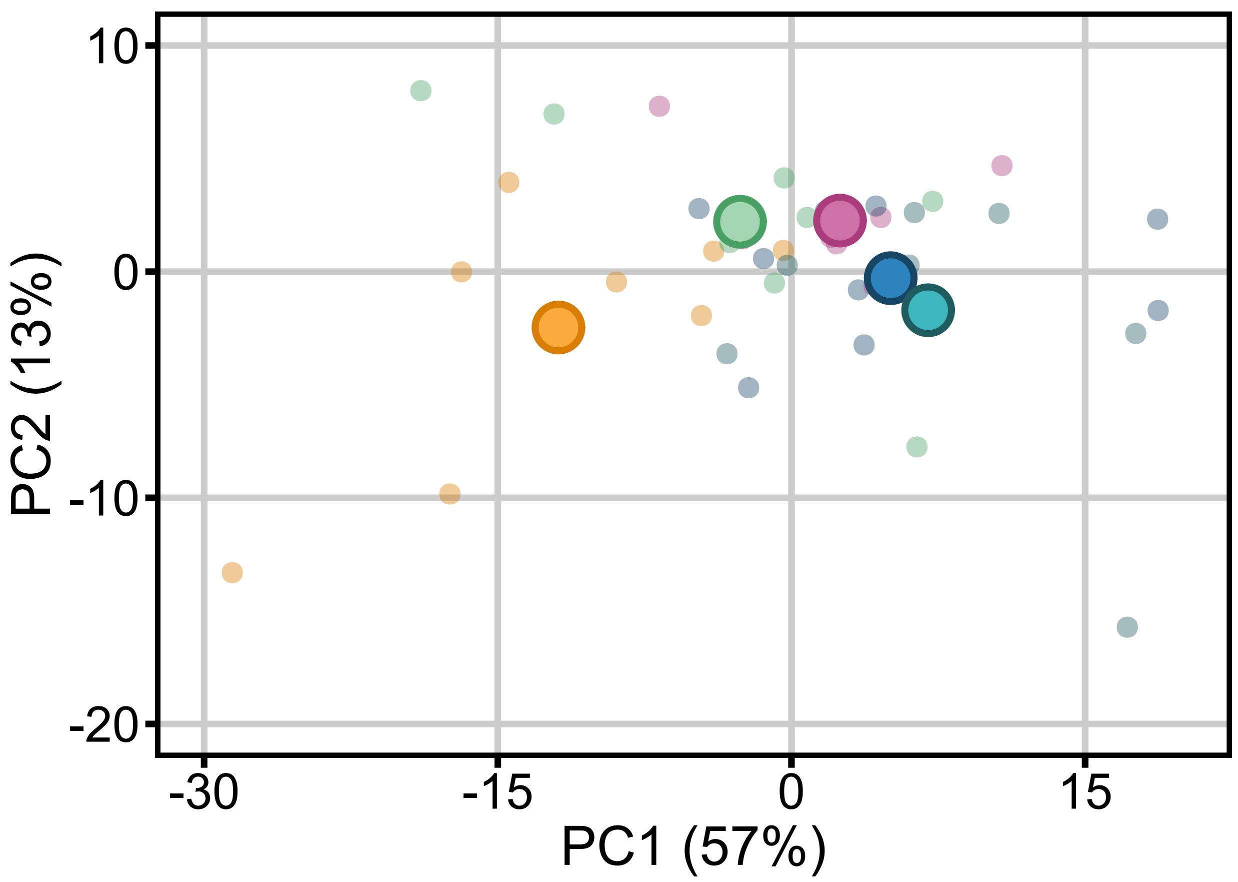
PCA plots

```
# PCA with mean summarisation
pca <- doPCA(RNAseq$filt$DESeq_vst_nobatch)

df <- pca$pcs %>%
  cbind(RNAseq$filt$design)

df2 <- df %>%
  group_by(time) %>%
  summarize(PC1 = mean(PC1),
            PC2 = mean(PC2),
            PC3 = mean(PC3),) %>%
  dplyr::rename(time2 = time)

ggplot() +
  geom_point(data = df, aes(x=PC1, y=PC2, color=time), shape=16, size=3, stroke=0, alpha=0.4) +
  geom_point(data = df2, aes(x=PC1, y=PC2, fill=time2), color="black", shape=23, size=6, stroke=1.5, alpha=1) +
  ggrepel::geom_label_repel() +
  xlab(paste("PC1 (", round(pca$percentVar[1],0), "%)", sep = "")) +
  ylab(paste("PC2 (", round(pca$percentVar[2],0), "%)", sep = "")) +
  scale_color_manual(values = colPals$time) +
  scale_fill_manual(values = colPals$time, name='Time') +
  guides(color = F, fill = guide_legend(override.aes = list(size=4))) +
  ggtitle('Volunteers 9 & 10 excluded') +
  theme_bw(base_size = 16) +
  theme(
    legend.position = 'right',
    legend.justification = 'top',
    plot.title = element_text(size=16, face='bold', hjust=0.5),
    axis.title = element_text(size=16, face='bold'),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.border = element_rect(color = "black", fill = NA, size = 2),
    axis.ticks = element_line(color = "black", size = 1.25)
  )
```



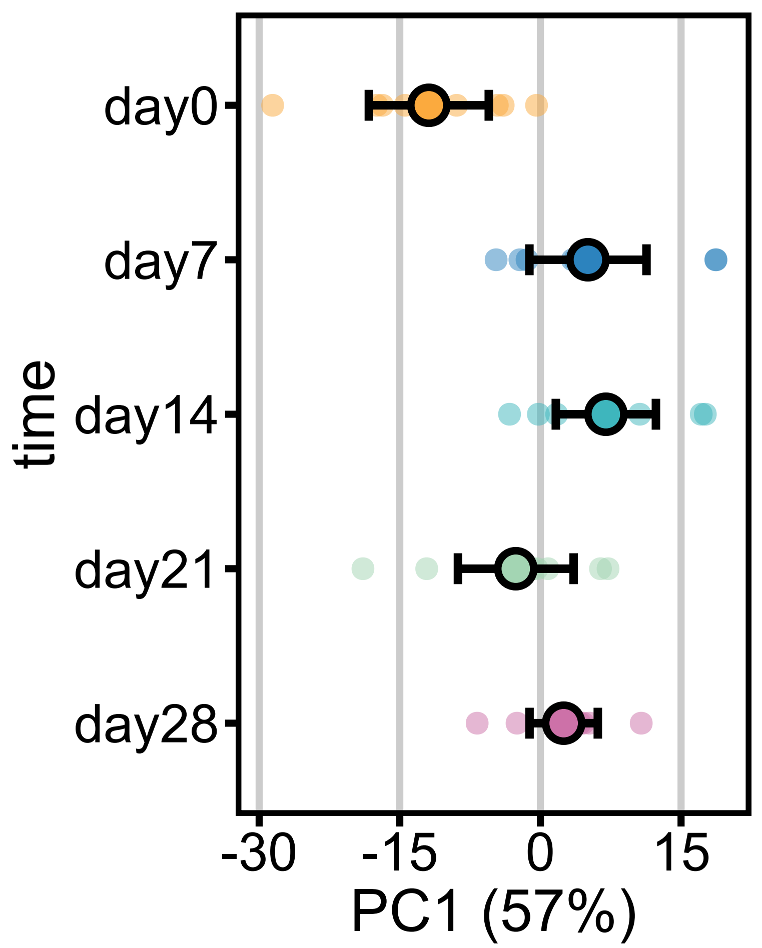
```
ggsave("plots/figS2_pca_volunteers_9&10_excl.pdf", width = 6, height = 4, units = "in", dpi = 300, device = cairo_pdf)
```

```
pca <- doPCA(RNAseq$filt$DESeq_vst_nobatch)

df <- pca$pcs %>%
  cbind(RNAseq$filt$design)

df2 <- df %>%
  group_by(time) %>%
  summarize(PC1 = mean(PC1),
            PC2 = mean(PC2),
            PC3 = mean(PC3),) %>%
  dplyr::rename(time2 = time)

ggplot() +
  geom_point(data = df, aes(x=PC1, y=PC2, color=time), shape=16, size=4, stroke=0, alpha=0.4) +
  geom_point(data = df2, aes(x=PC1, y=PC2, color=time2, fill=time2), shape=21, size=7.5, stroke=2, alpha=1) +
  ggrepel::geom_label_repel() +
  theme_custom(base_size = 20) +
  xlab(paste("PC1 (", round(pca$percentVar[1],0), "%)", sep = "")) +
  ylab(paste("PC2 (", round(pca$percentVar[2],0), "%)", sep = "")) +
  scale_color_manual(values = colPals$time_dark) +
  scale_fill_manual(values = colPals$time) +
  scale_x_continuous(limits = c(-30,20), breaks = seq(-30,15,15)) +
  scale_y_continuous(limits = c(-20,10),breaks = seq(-20,10,10)) +
  theme(panel.grid.major = element_line(color = "grey80", linetype = "solid", size = 1.25),
        panel.grid.minor = element_line(color = "transparent", linetype = "solid"),
        panel.border = element_rect(color = "black", fill = NA, size = 2),
        axis.ticks = element_line(color = "black", size = 1.25),
        legend.position = "none")
```



```
ggsave(filename = "plots/fig1D_pca_volunteers_9&10_excl.pdf", width = 7, height = 5, units = "in", dpi = 300, device = cairo_pdf)
```

```
df2 <- df %>%
  group_by(time) %>%
  select(time, PC1) %>%
  summarize_each(dplyr::funs(mean, sd, se=sd(.)/sqrt(n())), PC1) %>%
  dplyr::rename(time2 = time)

ggplot() +
  geom_point(data = df, aes(x=time, y=PC1, color=time), shape=16, size=4, stroke=0, alpha=0.5) +
  geom_errorbar(data=df2, aes(x=time2, y=mean, ymin=mean-se*1.96, ymax=mean+se*1.96), width=.2, lwd=1.5) +
  geom_point(data = df2, aes(x=time2, y=mean, fill=time2), color="black", shape=21, size=5, stroke=2, alpha=1) +
  scale_x_discrete(limits = rev(levels(df$time))) +
  scale_y_continuous(limits = c(-30,20), breaks = seq(-30,15,15)) +
  coord_flip() +
  theme_custom(base_size = 20) +
  ylab(paste("PC1 (", round(pca$percentVar[1],0), "%)", sep = "")) +
  scale_color_manual(values = colPals$time) +
  scale_fill_manual(values = colPals$time) +
  theme(panel.grid.major.x = element_line(color = "grey80", linetype = "solid", size = 1.25),
        panel.grid.minor = element_line(color = "transparent", linetype = "solid"),
        panel.border = element_rect(color = "black", fill = NA, size = 2),
        axis.ticks = element_line(color = "black", size = 1.25),legend.position = "none")
```

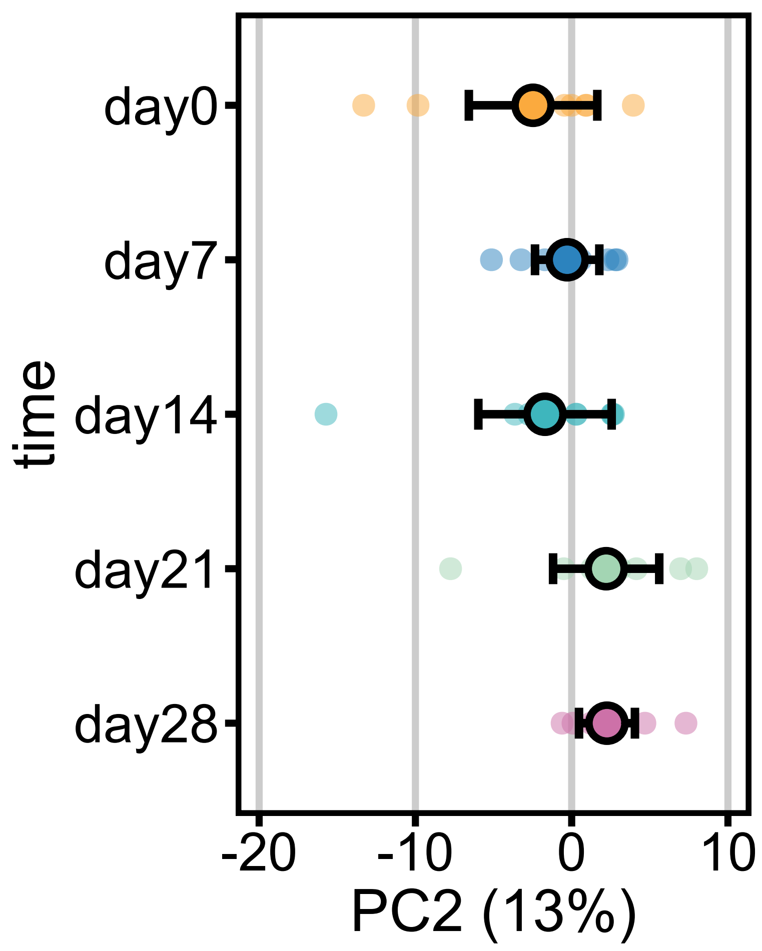


```
ggsave(filename = "plots/fig1E_pca_volunteers_9&10_excl_PC1.pdf", width = 4, height = 5, units = "in", dpi = 300, device = cairo_pdf)
```

```
df2 <- as.data.frame(pca$rotation) %>%
  rownames_to_column(var = "GeneSymbol") %>%
  select(GeneSymbol, PC1) %>%
  mutate(sign = sign(PC1)) %>%
  mutate(PC1_abs = abs(PC1)) %>%
  arrange(desc(PC1_abs))

df2$GeneSymbol[1:10]
```

```
##  [1] "SIK1B"   "NR4A2"   "SIK1"    "TNFAIP3" "RGS1"    "TENT5C"  "CSRNP1"
##  [8] "DUSP2"   "PER1"    "SOCS3"
```

```
df2 <- df %>%
  group_by(time) %>%
  select(time, PC2) %>%
  summarize_each(dplyr::funs(mean, sd, se=sd(.)/sqrt(n())), PC2) %>%
  dplyr::rename(time2 = time)

ggplot() +
  geom_point(data = df, aes(x=time, y=PC2, color=time), shape=16, size=4, stroke=0, alpha=0.5) +
  geom_errorbar(data=df2, aes(x=time2, y=mean, ymin=mean-se*1.96, ymax=mean+se*1.96), width=.2, lwd=1.5) +
  geom_point(data = df2, aes(x=time2, y=mean, fill=time2), color="black", shape=21, size=5, stroke=2, alpha=1) +
  scale_x_discrete(limits = rev(levels(df$time))) +
  scale_y_continuous(limits = c(-20,10), breaks = seq(-20,10,10)) +
  coord_flip() +
  theme_custom(base_size = 20) +
  ylab(paste("PC2 (", round(pca$percentVar[2],0), "%)", sep = "")) +
  scale_color_manual(values = colPals$time) +
  scale_fill_manual(values = colPals$time) +
  theme(panel.grid.major.x = element_line(color = "grey80", linetype = "solid", size = 1.25),
        panel.grid.minor = element_line(color = "transparent", linetype = "solid"),
        panel.border = element_rect(color = "black", fill = NA, size = 2),
        axis.ticks = element_line(color = "black", size = 1.25),legend.position = "none")
```



```
df2 <- as.data.frame(pca$rotation) %>%
  rownames_to_column(var = "GeneSymbol") %>%
  select(GeneSymbol, PC2) %>%
  mutate(sign = sign(PC2)) %>%
  mutate(PC2_abs = abs(PC2)) %>%
  arrange(desc(PC2_abs))

df2$GeneSymbol[1:11]
```

```
##  [1] "LYZ"             "ENSG00000257764" "FCN1"            "S100A9"
##  [5] "S100A8"          "VCAN"            "CSF3R"           "SERPINA1"
##  [9] "TNFAIP2"         "MPEG1"           "IFI30"
```

Exports

```
saveRDS(RNAseq, file= "data/rnaseq/rnaseq_volunteers_9&10_excl.rds")
saveRDS(DESeq2_DEGs, file= "data/rnaseq/DESeq2_DEGs_unfilt_volunteers_9&10_excl.rds")
saveRDS(DESeq2_DEGs_filt, file= "data/rnaseq/DESeq2_DEGs_filt_volunteers_9&10_excl.rds")

openxlsx::write.xlsx(DESeq2_DEGs_filt[2:5], file = "tables/dataS3_DEGs_filtered.xlsx", rowNames=F, overwrite=T)
```

SessionInfo

```
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=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] RColorBrewer_1.1-3          variancePartition_1.28.9
##  [3] BiocParallel_1.32.6         limma_3.54.2
##  [5] DESeq2_1.38.3               SummarizedExperiment_1.28.0
##  [7] Biobase_2.58.0              MatrixGenerics_1.10.0
##  [9] matrixStats_0.63.0          GenomicRanges_1.50.2
## [11] GenomeInfoDb_1.34.9         IRanges_2.32.0
## [13] S4Vectors_0.36.2            BiocGenerics_0.44.0
## [15] patchwork_1.1.2             magrittr_2.0.3
## [17] forcats_1.0.0               stringr_1.5.0
## [19] dplyr_1.1.1                 purrr_1.0.1
## [21] readr_2.1.4                 tidyr_1.3.0
## [23] tibble_3.2.1                ggplot2_3.4.2
## [25] tidyverse_1.3.2
##
## loaded via a namespace (and not attached):
##   [1] googledrive_2.1.0       minqa_1.2.5             colorspace_2.1-0
##   [4] XVector_0.38.0          fs_1.6.1                rstudioapi_0.14
##   [7] farver_2.1.1            ggrepel_0.9.3           bit64_4.0.5
##  [10] mvtnorm_1.1-3           AnnotationDbi_1.60.2    fansi_1.0.4
##  [13] lubridate_1.9.2         xml2_1.3.3              codetools_0.2-18
##  [16] splines_4.2.1           doParallel_1.0.17       cachem_1.0.7
##  [19] geneplotter_1.76.0      knitr_1.42              jsonlite_1.8.4
##  [22] nloptr_2.0.3            pbkrtest_0.5.2          RhpcBLASctl_0.23-42
##  [25] broom_1.0.4             annotate_1.76.0         dbplyr_2.3.2
##  [28] png_0.1-8               aod_1.3.2               compiler_4.2.1
##  [31] httr_1.4.5              backports_1.4.1         Matrix_1.5-3
##  [34] fastmap_1.1.1           gargle_1.3.0            cli_3.6.1
##  [37] prettyunits_1.1.1       htmltools_0.5.5         tools_4.2.1
##  [40] gtable_0.3.3            glue_1.6.2              GenomeInfoDbData_1.2.9
##  [43] reshape2_1.4.4          clusterGeneration_1.3.7 Rcpp_1.0.10
##  [46] cellranger_1.1.0        vctrs_0.6.1             Biostrings_2.66.0
##  [49] nlme_3.1-157            iterators_1.0.14        remaCor_0.0.11
##  [52] xfun_0.38               rbibutils_2.2.13        openxlsx_4.2.5.1
##  [55] lme4_1.1-32             rvest_1.0.3             timechange_0.2.0
##  [58] lifecycle_1.0.3         gtools_3.9.4            XML_3.99-0.14
##  [61] googlesheets4_1.1.0     zlibbioc_1.44.0         MASS_7.3-57
##  [64] scales_1.2.1            hms_1.1.3               parallel_4.2.1
##  [67] yaml_2.3.7              memoise_2.0.1           stringi_1.7.12
##  [70] RSQLite_2.3.0           highr_0.10              foreach_1.5.2
##  [73] caTools_1.18.2          zip_2.2.2               boot_1.3-28
##  [76] Rdpack_2.4              rlang_1.1.0             pkgconfig_2.0.3
##  [79] bitops_1.0-7            evaluate_0.20           lattice_0.20-45
##  [82] labeling_0.4.2          bit_4.0.5               tidyselect_1.2.0
##  [85] plyr_1.8.8              R6_2.5.1                gplots_3.1.3
##  [88] generics_0.1.3          RUnit_0.4.32            DelayedArray_0.24.0
##  [91] DBI_1.1.3               pillar_1.9.0            haven_2.5.2
##  [94] withr_2.5.0             KEGGREST_1.38.0         RCurl_1.98-1.12
##  [97] modelr_0.1.11           crayon_1.5.2            KernSmooth_2.23-20
## [100] utf8_1.2.3              tzdb_0.3.0              rmarkdown_2.21
## [103] progress_1.2.2          locfit_1.5-9.7          grid_4.2.1
## [106] readxl_1.4.2            blob_1.2.4              reprex_2.0.2
## [109] digest_0.6.31           xtable_1.8-4            munsell_0.5.0
```