R Code Supplement

R code to process Whole Blood Microarrays after Stimulation (chitin and others)

This report uses various R packages to walk the reader from the raw data to the paper's main figures. It is intended to create full transparency and computational reproducibility, and further offers deeper insights into the data than possible in the main figure. We welcome any feedback and invite you to keep it friendly and constructive.

This report was compiled by Felix Frauhammer (whose personal opinion is that all published papers should include the code of their computational analysis) using R function `knitr::spin("file",knit=F)` and RStudio's `Compile Report` button to knit the Rmd into PDF.

```
date() # date of compilation
```

```
## [1] "Mon Mar  5 14:47:18 2018"
```

This report goes together with the following paper:

- Whole blood gene expression after stimulation with TLR2 ligands (chitin oligomers, Pam2, Pam3) and the TLR4-ligand LPS
- Corresponding author: Alexander NR Weber (E-mail alexander.weber@uni-tuebingen.de)
- GEO record GSE103094 (download the GSE103094_non-normalized.txt.gz and unzip it to get started)

```
#
```

Information on the 24 microarrays in this dataset:

It has the following treatments:

- Unstimulated
- PAM2: Pam2CSK4 (TLR2 ligand)
- PAM3: Pam3CSK4 (TLR2 ligand)
- Chit: chitin oligomers (object of this study)
- LPS: lipopolysaccharide (TLR4 ligand)

```
#
```

It has the following donors:

- P1: 4 treatments
- P2: 4 treatments
- P3: 5 treatments
- P4: 5 treatments
- P5: 5 treatments

```
#
```

Install these packages if not already available on your machine:

```
library(limma)
library(ggplot2)
library(cowplot)
library(tidyverse)
library(pheatmap)
library(ComplexHeatmap)
# see end of script for the package versions.
```

Read in data

Option1: raw IDAT files

While we recommend using the next paragraph ("`Option2: GSE103094_non-normalized.txt`"), here are useful hints as to how you would process the raw files.

This data was generated using the **'Illumina Arrays HumanHT-12v4'**, and the Illumina BeadScan software outputs IDAT-files. In order to read these into R, you need to download the ~ 70 MB Manifest File "HumanHT-12 v4.0 Manifest File (TXT Format)" from the chip's product files page, next to the raw array files from GEO, and then do this:

```
#    idatfiles.all <- dir(pattern="idat") # in download directory
#    # idatfiles.all should now be a character vector with IDAT-filepaths
#    chitin.raw <- limma::read.idat(idatfiles.all, bgxfile="path/to/manifest-file.bgx")
#    # should produce the same as `chitin.raw` produced in next paragraph
```

Option2: GSE103094_non-normalized.txt

We assume you have downloaded and unzipped the supplementary file from our GEO Record. Then you can read it into R like this:

```
fileFromGeo <- "/home/felix/chitin_microarrays/data/GSE103094_non-normalized.txt"
               # adapt path to where you downloaded the file
GEOdata <- read.csv(fileFromGeo, header=T, sep="\t" )
```

Quick overview over this table:

- The Column "Status" lists the control probes (ERCCs, negative controls, etc.) and then the 47323 probes measuring gene expression.
- each sample has two columns, one with gene expression and one with the DetectionPvalue, i.e. whether the gene in this array is significantly above background signal
- The other three columns are Illumina "Probe_Id", "Array_Address_Id" and, most useful for the biology, the Gene Symbol.

```
#
```

Convert to limma class "EListRaw"

```
# extract expression columns:
geneExprs <- as.matrix(GEOdata[,-c(1:4)])
geneExprs <- geneExprs[,!grepl("Detection", colnames(geneExprs))]
# Create EListRaw object. Note that the neqc function below uses the information in
# column `Status` of the the genes-slot to know which probes are "negative" and which are
# "regular" (i.e. measured Genes) for background correction, etc..
chitin.raw <- new("EListRaw",list(E=geneExprs, genes=GEOdata[,1:4],
     other=list(Detection=GEOdata[,grepl("Detection", colnames(GEOdata))])))
```

Normalization, filtering and sample Information

We use limma's robust neqc-function for normalization, which completes the following steps for us:

- corrects background (as measured by negative probes, provided in slot chitin.raw*genes*Status)
- removes control probes (and the now obsolete slot genes$Status)
- log2-transforms data (see further: `limma::limmaUsersGuide()`, section 17.3)

```
chitin <- neqc(chitin.raw)
```

Next, we filter the measured genes by removing non-expressed probes. We only keep genes detected significantly ($p<.05$) above background in more than 3 arrays:

```
expressed <- rowSums(chitin$other$Detection < 0.05)  > 3
chitin <- chitin[expressed,]  # filters out ~ 24,000 probes
```

For analysis below, we want to know the Treatment and Patient identiy of each array. It's part of the colnames, so we can extract it conveniently here:

```
meta <- data.frame(t(rbind.data.frame( strsplit(colnames(chitin), "_"))),
                   row.names=colnames(chitin))
colnames(meta) <-c("Treatment","Patient")
```

Principal Component Analysis

Below we compute principal component analysis on all non-filtered probes (22929 genes) and then again on 6191 genes that vary significantly between treatment groups according to ANOVA (done with limma's F-test). In the paper's main figure, we show the latter as the former is influenced more by patient-patient heterogeneity, as we'll see below.

```
#
```

F-test for high-variance genes

This paragraph identifies genes that vary significantly across treatment groups (Chitin, LPS, PAM2, PAM3, Unstimulated). The 6191 identified genes and their expression values (see object `anovaexprs` at end of paragraph) will go into PCA, see next heading.

```
# design matrix for lmFit
ct <- factor(meta$Treatment)
# It is good practice to set the control as first level:
ct <- relevel(ct, "Unstimulated")
design <- model.matrix(~0+ct)
colnames(design) <- levels(ct)
# duplicate correlation for lmFit
dupcor <- duplicateCorrelation(chitin, design, block=meta$Patient)
dupcor$consensus.correlation  # can be fed into lmFit function
```

```
## [1] 0.3112271
```

We see that differently treated samples from the same patient correlate moderately with each other (~0.3), as expected from human material. Notifying `lmFit` of this patient-patient heterogeneity will make the analysis more robust.

```
# fit limma model:
fit <- lmFit(chitin, design, block=meta$Patient,
             correlation = dupcor$consensus.correlation)

contrasts.all <- makeContrasts(
        Chitin-LPS,Chitin-PAM2, Chitin-PAM3, Chitin-Unstimulated,
        LPS-PAM2, LPS-PAM3, LPS-Unstimulated,
        PAM2-PAM3,PAM2-Unstimulated, PAM3-Unstimulated,
        levels=design
)
fit2.all <- contrasts.fit(fit, contrasts.all)
fit2.all <- eBayes(fit2.all, trend=T)
```

The two lines above compute an F-test, which is equivalent to a one-way ANOVA for each gene except that the residual mean squares have been moderated between genes. (see `limma::limmaUsersGuide()`) After multiple testing correction with Benjamini-Hochberg, 6191 genes are found to vary across different groups with FDR<.05:

```
anova <- topTableF(fit2.all, p.value=0.05, number=50000, adjust.method = 'BH')
dim(anova)
```

```
## [1] 6191   17
```

These genes contain the information on what separates different treatments independent of the patient of origin. To analyze them in PCA, we extract the expression values of these genes:

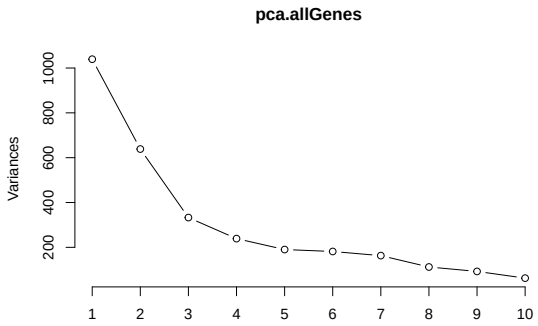
```
anovaexprs <- chitin$E
rownames(anovaexprs) <- chitin$genes$Probe_Id
anovaexprs <- anovaexprs[rownames(anovaexprs) %in% anova$Probe_Id,]
```

PCA on all genes separates Patients

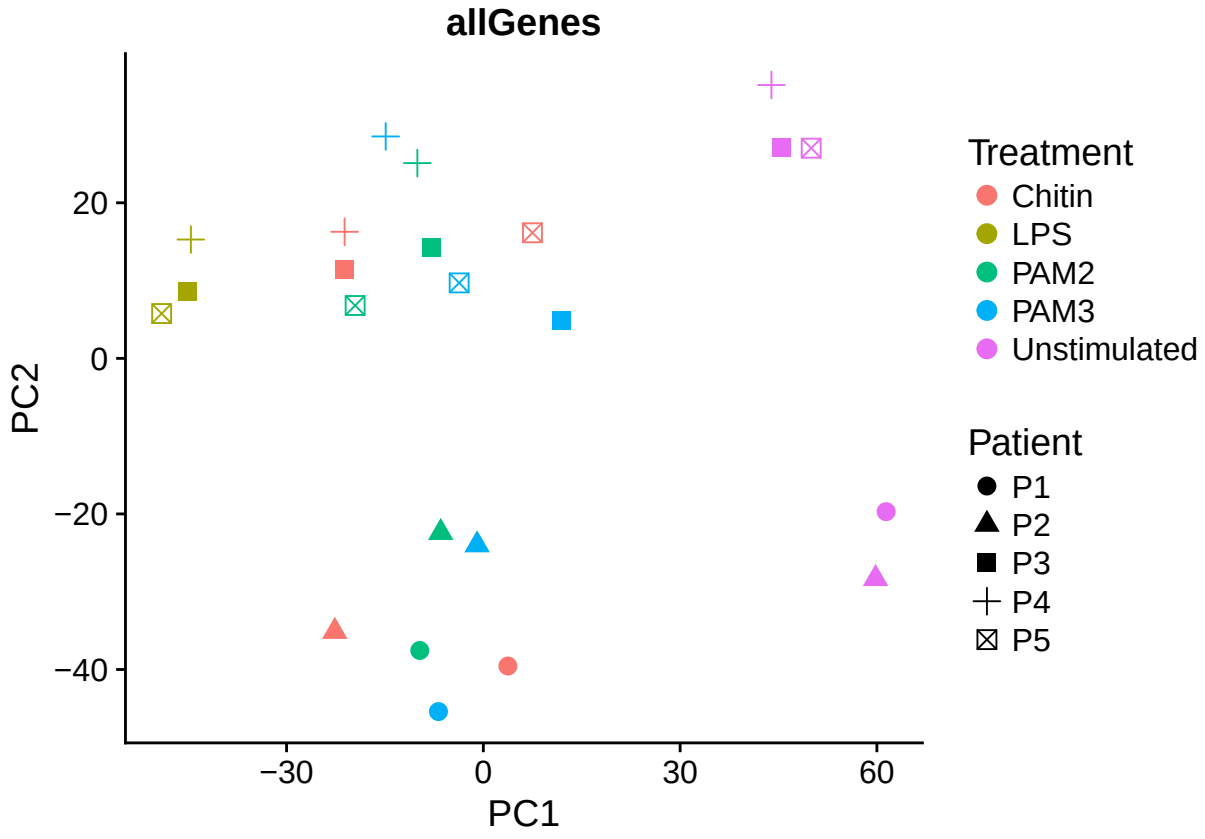
Human material often shows high patient-patient variability. In line with this, the largest variation captured by PCA on all genes is driven by patient-patient differences, as we see in the following.

```
pca.allGenes <- prcomp(t(chitin$E))
scores.allGenes <- cbind(as.data.frame(pca.allGenes$x), meta)
```

```
screeplot(pca.allGenes, type = "lines")
```




```
ggplot(scores.allGenes, aes(x=PC1, y=PC2, col=Treatment, shape=Patient))+
  geom_point(size=3)+ggtitle("allGenes")
```



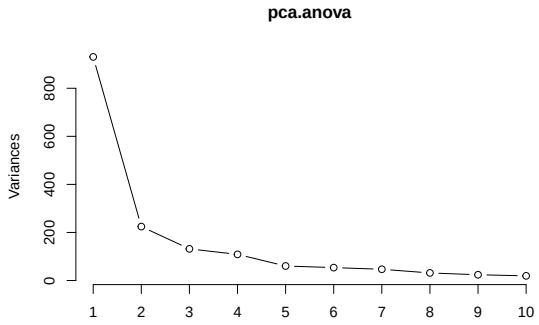
We see that when computed on all genes, PC1 indicates a strong difference between Unstimulated, LPS and the others, while PC2 separates Patients P1 and P2 from other donors. To better resolve the differences of Chitin, PAM2 and PAM3, we compute PCA on genes that vary between groups in the next paragraph, hoping to reduce how much PCA is influenced by donor heterogeneity.

```
#
```

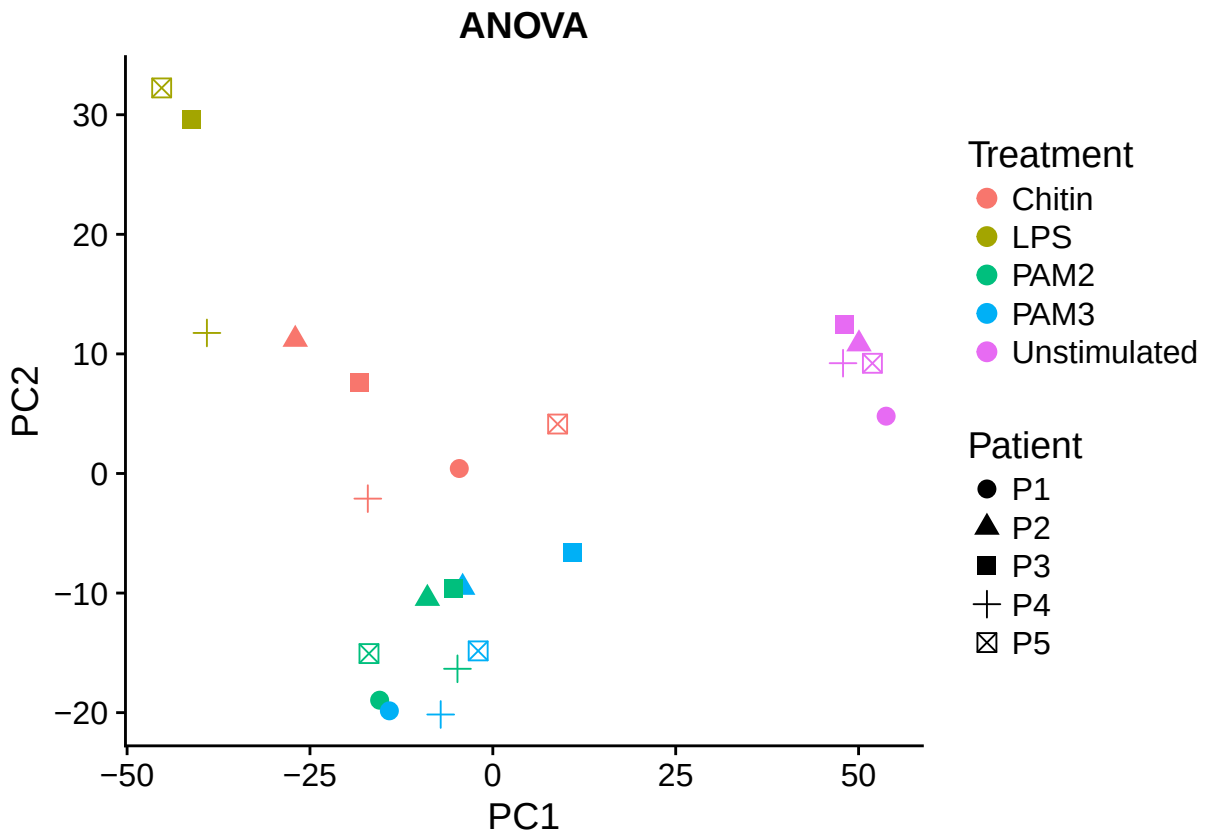
PCA on 6191 ANOVA genes

```
pca.anova    <- prcomp(t(anovaexprs))
scores.anova <- cbind(as.data.frame(pca.anova$x), meta)
```

```
screeplot(pca.anova, type = "lines")
```




```
ggplot(scores.anova, aes(x=PC1, y=PC2, col=Treatment, shape=Patient))+
  geom_point(size=3)+ggtitle("ANOVA")
```



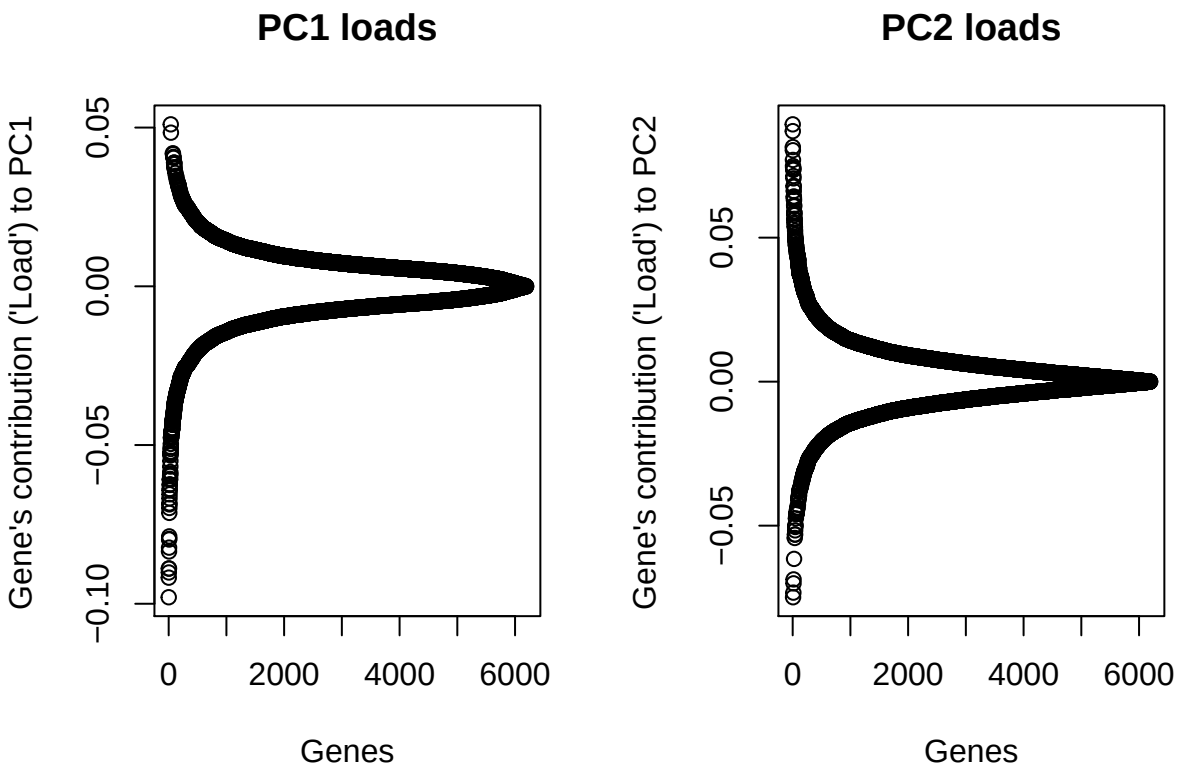
As hoped, the samples do not cluster by patient anymore. As above, PC1 separates Unstimulated and LPS from the others. Interestingly, PAM2 and PAM3 seem very similar to each other, while all chitin samples appear clearly distinct from PAM and LPS along PC2. We note, however, that above scree-plot shows PC2 to play a rather minor role, as most variance in the data is captured by PC1. This already indicates that chitin, PAM2 and PAM3 show rather minor differences relative to the effects induced by LPS. For biological interpretation of this PCA, we will look into the genes contributing to PCs 1 and 2 in the next subparagraph, while the next big heading will find and display Differentially Expressed Genes between treatment groups.

```
#
```

Obtain and export PCA loadings

To place the PCA figure in above's paragraph into biological context, we extract the information on which genes contribute the most to PC1 and PC2.

```
par(mfrow=c(1,2))
plot(pca.anova$rotation[,1][order(abs(pca.anova$rotation[,1]), decreasing=T)],
     main="PC1 loads", ylab="Gene's contribution ('Load') to PC1", xlab="Genes")
plot(pca.anova$rotation[,2][order(abs(pca.anova$rotation[,2]), decreasing=T)],
     main="PC2 loads", ylab="Gene's contribution ('Load') to PC2", xlab="Genes")
```




```
par(mfrow=c(1,1))
```

Notes on above plot: Genes with large absolute loads contribute most to PCs. For example, the genes with loads around 0.05 make a sample's PC1 score positive (e.g. all Unstimulated samples in the "ANOVA"-PCA plot above), whereas high expression of the negative-load genes makes a sample show negative PC1 values (e.g. LPS samples in above plot).

From the plots above we can estimate roughly by eye that perhaps the 100 or 300 genes with the highest load are interesting for biological interpretation, as it is mainly their cumulative load separating the different treatments from each other in PCA plot "ANOVA". Still, we export all genes that went into the PCA, for completeness:

```
topPC1 <- pca.anova$rotation[,1, drop=F]
topPC1 <- topPC1[order(abs(topPC1), decreasing=T),,drop=F]
topPC2 <- pca.anova$rotation[,2, drop=F]
topPC2 <- topPC2[order(abs(topPC2), decreasing=T),,drop=F]
# bring it all into same dataframe and add gene Symbol annotations:
PCtopDF <- data.frame(PC1_load=topPC1[,1], PC1_ProbeID=rownames(topPC1),
           PC1_Symbol=GEOdata$Symbol[match(rownames(topPC1), GEOdata$Probe_Id)],
           PC2_load=topPC2[,1], PC2_ProbeID=rownames(topPC2),
           PC2_Symbol=GEOdata$Symbol[match(rownames(topPC2), GEOdata$Probe_Id)],
           stringsAsFactors = F
           )
# for duplicated symbols (whenever the array measures genes with >1 probes), we
# only keep the first probe to avoid confusion.
PCtopDF <- PCtopDF[!duplicated(PCtopDF$PC1_Symbol),]
```

```
PCtopDF <- PCtopDF[!duplicated(PCtopDF$PC2_Symbol),]
head(PCtopDF)
```

```
##                 PC1_load  PC1_ProbeID PC1_Symbol    PC2_load  PC2_ProbeID
## ILMN_1699651 -0.09794445 ILMN_1699651        IL6  0.08933929 ILMN_1707695
## ILMN_1671509 -0.09177125 ILMN_1671509       CCL3  0.08698045 ILMN_2054019
## ILMN_1747355 -0.09014691 ILMN_1747355     CCL3L1  0.08145545 ILMN_1701789
## ILMN_1805410 -0.08884788 ILMN_1805410   C15orf48  0.07698123 ILMN_1674811
## ILMN_1838319 -0.08351726 ILMN_1838319  LOC730249  0.07488550 ILMN_2067890
## ILMN_1720048 -0.08231378 ILMN_1720048       CCL2 -0.07485910 ILMN_1686116
##              PC2_Symbol
## ILMN_1699651      IFIT1
## ILMN_1671509      ISG15
## ILMN_1747355      IFIT3
## ILMN_1805410       OASL
## ILMN_1838319     CXCL11
## ILMN_1720048      THBS1
```

The first 3 columns rank genes according to their influence (load) on PC1, the last 3 for PC2. We now write out the complete table and provide it as supplementary table attached to this paper. Note that it includes all genes that vary across conditions (according to ANOVA), and the first few hundred genes in each column contribute the most to PCs 1 and 2.

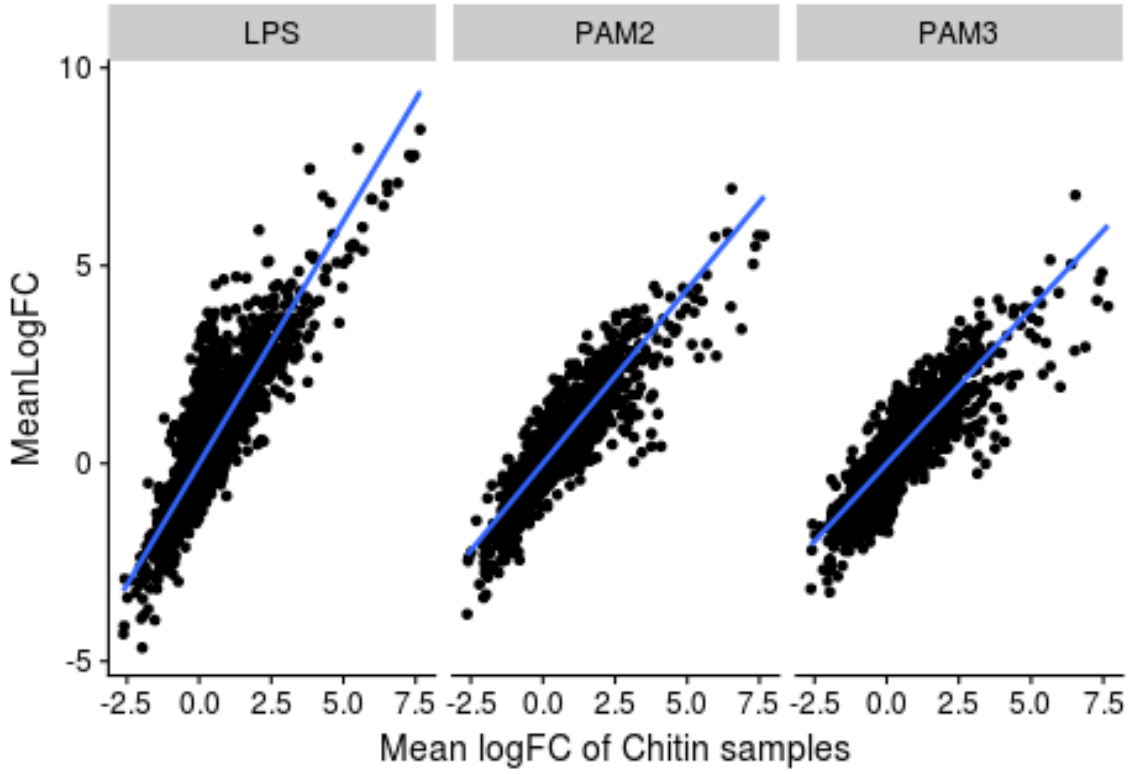
```
write_csv(x = PCtopDF, path = paste0("/home/felix/chitin_microarrays/PC1-2_topGenes",
                                     format(Sys.time(), "_%Y%m%d-%H.%M"),".csv"))
```

Heatmap for the main figure

The heatmap in the main figure is generated at the end of this section and meant to illustrate and emphasise the existing differences between Chitin and other treatments. It is, however, important to note at this point that in fact most genes up/down-regulated by one treatment are up/down-regulated in most other treatments as well. This is expected, as all treatments activate large immune signaling cascades. To make this fact clear to the reader, we now quickly compare the log-foldchanges of different treatments with each other, before moving on to the main heatmap.

```
# get logFCs for each sample ("relative expression", relExprs)
  relExprs <-NULL
  for(patient in levels(meta$Patient)){
    relExprs <-cbind(relExprs, apply(chitin$E[,
                        grepl(patient, colnames(chitin$E))],2, function(column)
                        column-chitin$E[,paste0("Unstimulated_",patient)]))
  }
  relExprs <- relExprs[,!grepl("Unstimulated", colnames(relExprs))]
# average logFCs per treatment:
  meanLogFC <- NULL
  Ts <-c("Chitin","LPS",  "PAM2","PAM3")
  for(treatment in Ts){
    meanLogFC <- cbind(meanLogFC,
                    rowMeans(relExprs[,grepl(treatment, colnames(relExprs))]))
  }
  colnames(meanLogFC) <- Ts
```

```
gather(as.data.frame(meanLogFC),key="Treatment", value="MeanLogFC", LPS:PAM3) %>%
  ggplot(aes(x=Chitin, y=MeanLogFC))+geom_point()+facet_wrap(~Treatment)+
  xlab("Mean logFC of Chitin samples")+ geom_smooth(method =lm)
```



As we see, whenever chitin-treatment induces a high log-foldchange for a gene, the other treatments do also lead to upregulation. One thing to note is that LPS has particularly many upregulations that do not change for chitin samples (left panel, note the many dots at x=0 lying far above the blue line), and LPS also has the steepest line, indicating all genes are upregulated more than for other treatments. Again, this is not unexpected, as we activate strong immune responses with all treatments. Still, we wanted to make this point clear so that the heatmap in the main paper figure can be read in the right context.

```
#
```

Find chitin-specific genes

Now that we are aware how all treatments induce similar big transcriptional changes, it is interesting to work out differences between them. For this, we will compute differentially expressed genes between Chitin and the other treatments, and display them in a heatmap for the main paper.

```
# chitin-specific genes (higher/lower than all other treatments)
contrasts.cs <- makeContrasts(
        Chitin-PAM2,Chitin-PAM3, Chitin-LPS,
        levels=design)
```

Firstly, we are interested in genes where chitin is expressed highest. These are not too many, as we just saw that LPS is the most potent inducer of gene expression, so we choose a rather loose cutoff of foldchange=1.1 to be able to look at a few:

```
lfc1.1 <- decideTests(treat( contrasts.fit(fit, contrasts.cs), lfc = log2(1.1) ) )
chit_gt_all <- chitin[rowSums(lfc1.1)==3,]
dim(chit_gt_all)
```

```
## [1] 11 23
```

11 genes are 10 % or more higher in chitin than all three other treatments.

```
#
```

Other than being strictly highest, we next ask for genes where Chitin is the most different from the other Treatments. For this we identify genes that show misregulation compared to both PAMs (foldchange 1.2 or higher), irrespective of whether it is up- or downregulated. For a better overview in the final heatmap, we choose to split the resulting genes into two groups: one where chitin and LPS show no sign. difference (chit_LPSnoChange) and one where they do (chit_LPS_DE).

```
lfc1.2<- decideTests(treat( contrasts.fit(fit, contrasts.cs), lfc = log2(1.2) ) )
# identify genes significantly up- OR downregulated between chitin and PAM2 OR PAM3.
   chit_LPSnoChange <- chitin[ (lfc1.2[,1]!=0 & lfc1.2[,2]!=0) & lfc1.2[,3]==0 , ]
   chit_LPS_DE <- chitin[ (lfc1.2[,1]!=0 & lfc1.2[,2]!=0) & lfc1.2[,3]!=0 , ]
   # to avoid confusion, if multiple array probes measured the same gene we only keep one:
   chit_LPSnoChange <- chit_LPSnoChange[!duplicated(chit_LPSnoChange$genes$Symbol),]
   chit_LPS_DE <- chit_LPS_DE[!duplicated(chit_LPS_DE$genes$Symbol),]
   # we also want to display each gene only once in the heatmap:
   chit_LPS_DE <- chit_LPS_DE[! chit_LPS_DE$genes$Symbol %in% chit_gt_all$genes$Symbol,]
   chit_LPSnoChange <- chit_LPSnoChange[! chit_LPSnoChange$genes$Symbol %in%
                                          chit_gt_all$genes$Symbol,]
```

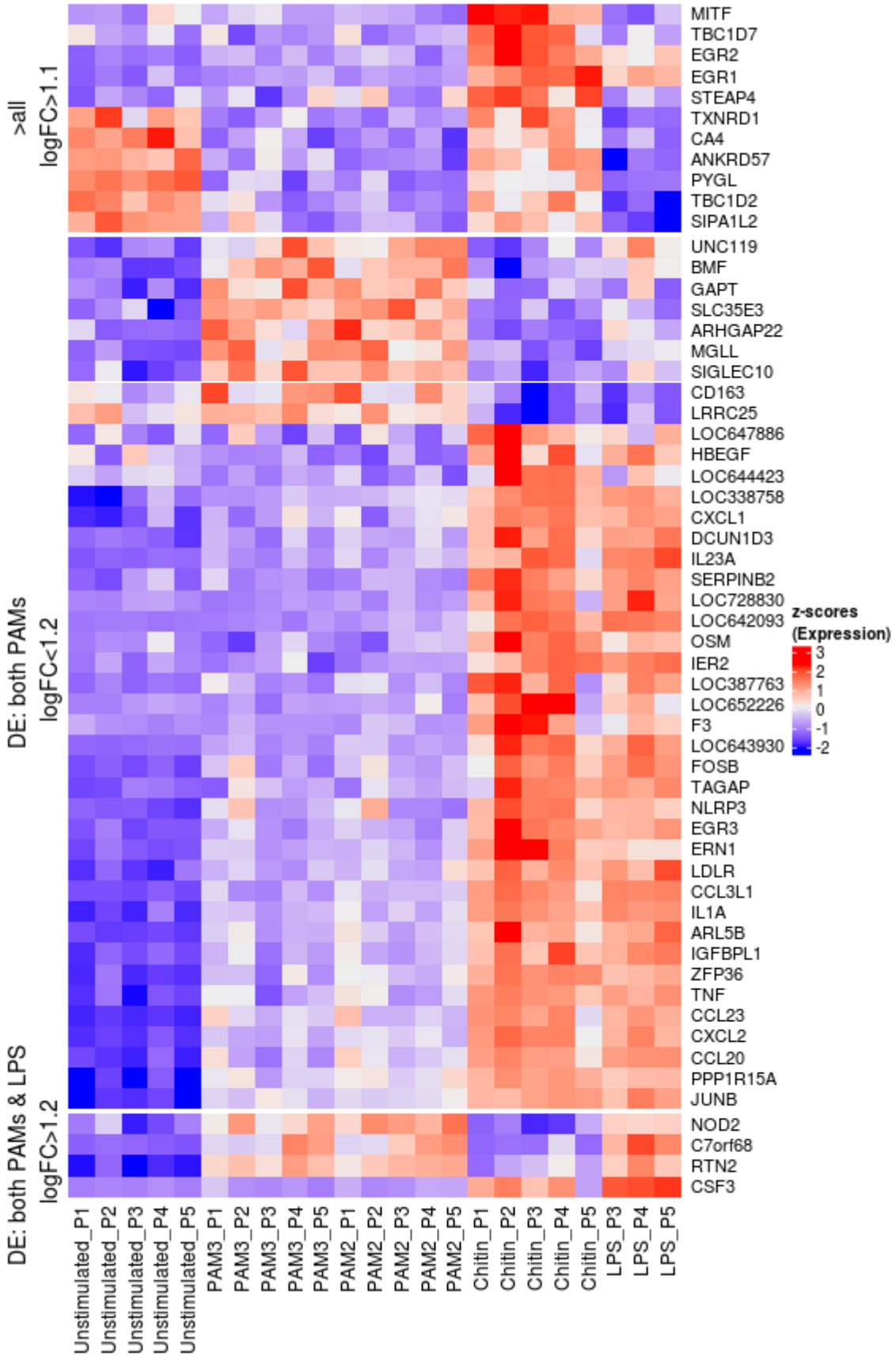
Create Heatmap from main figure

In the above paragraph we have selected interesting genes to look at, and now display them in a heatmap.

```
# We first sort the columns of the expression matrix by treatments:
heatmap.E <- data.frame(Symbol=chitin$genes$Symbol,
   chitin$E[,grepl("Unstimulated", colnames(chitin$E))],
   chitin$E[,grepl("PAM3",colnames(chitin$E))],
   chitin$E[,grepl("PAM2", colnames(chitin$E))],
   chitin$E[,grepl("Chitin",colnames(chitin$E))],
   chitin$E[,grepl("LPS", colnames(chitin$E))])
  rownames(heatmap.E) <- chitin$genes$Probe_Id
# we now extract the genes of interest, standardize them with `scale` (to get
# z-scores) and assign the corresponding gene Symbols:
genesOfInterest<-c(as.character(chit_gt_all$genes$Probe_Id),
                   as.character(chit_LPS_DE$genes$Probe_Id),
                   as.character(chit_LPSnoChange$genes$Probe_Id))
  hm <- heatmap.E[ match(genesOfInterest, rownames(heatmap.E)), ]
  rownames(hm) <- hm$Symbol; hm$Symbol <- c()
  hm <- data.frame(t(scale(t(hm))) )
```

We create the heatmap as in the main figure using the `ComplexHeatmap` package. Note that we deactivate clustering columns on purpose (while clustering is a great approach for unsupervised analysis, using it here would defeat its purpose: the heatmap should visualize genes selected by their differential expression, which is a supervised approach).

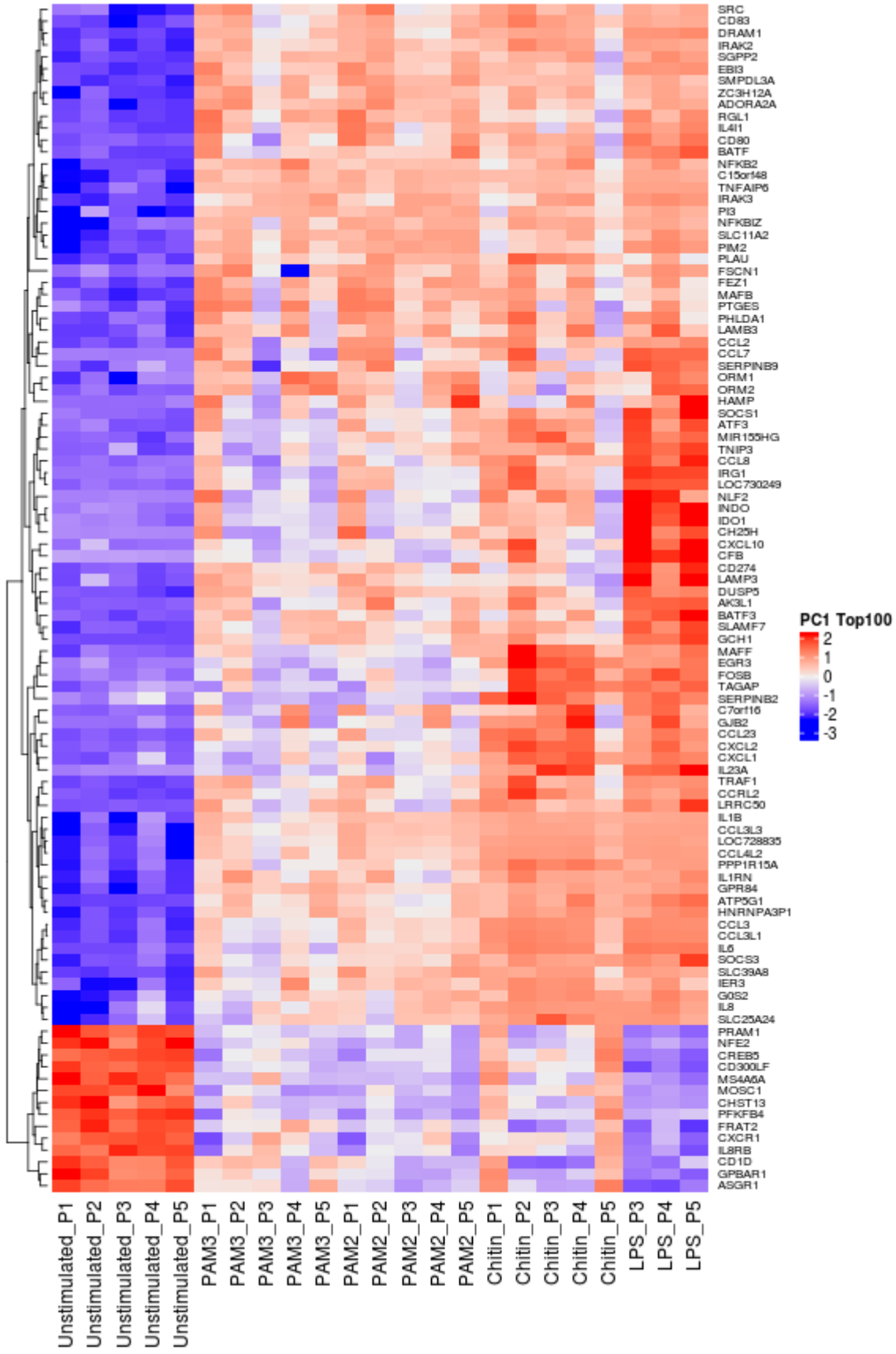
```
Heatmap(hm, cluster_columns = F, cluster_rows = T,
        show_row_dend=F,
                   split=c(
                        rep(">all\n logFC>1.1",dim(chit_gt_all)[1]),
                        rep("DE: both PAMs & LPS\n logFC>1.2",dim(chit_LPS_DE)[1]),
                        rep("DE: both PAMs\n logFC<1.2",dim(chit_LPSnoChange)[1])  ),
        name=paste("z-scores","(Expression)", sep = "\n"), row_names_gp = gpar(fontsize=11)
        )
```


>all
logFC>1.1
DE: both PAMs
logFC<1.2
DE: both PAMs & LPS
logFC>1.2
MITF
TBC1D7
EGR2
EGR1
STEAP4
TXNRD1
CA4
ANKRD57
PYGL
TBC1D2
SIPA1L2
UNC119
BMF
GAPT
SLC35E3
ARHGAP22
MGLL
SIGLEC10
CD163
LRRC25
LOC647886
HBEGF
LOC644423
LOC338758
CXCL1
DCUN1D3
IL23A
SERPINB2
LOC728830
LOC642093
OSM
IER2
LOC387763
LOC652226
F3
LOC643930
FOSB
TAGAP
NLRP3
EGR3
ERN1
LDLR
CCL3L1
IL1A
ARL5B
IGFBPL1
ZFP36
TNF
CCL23
CXCL2
CCL20
PPP1R15A
JUNB
NOD2
C7orf68
RTN2
CSF3
z-scores
(Expression)
3
2
1
0
-1
-2
Unstimulated_P1
Unstimulated_P2
Unstimulated_P3
Unstimulated_P4
Unstimulated_P5
PAM3_P1
PAM3_P2
PAM3_P3
PAM3_P4
PAM3_P5
PAM2_P1
PAM2_P2
PAM2_P3
PAM2_P4
PAM2_P5
Chitin_P1
Chitin_P2
Chitin_P3
Chitin_P4
Chitin_P5
LPS_P3
LPS_P4
LPS_P5

Supplementary figure: PC top genes

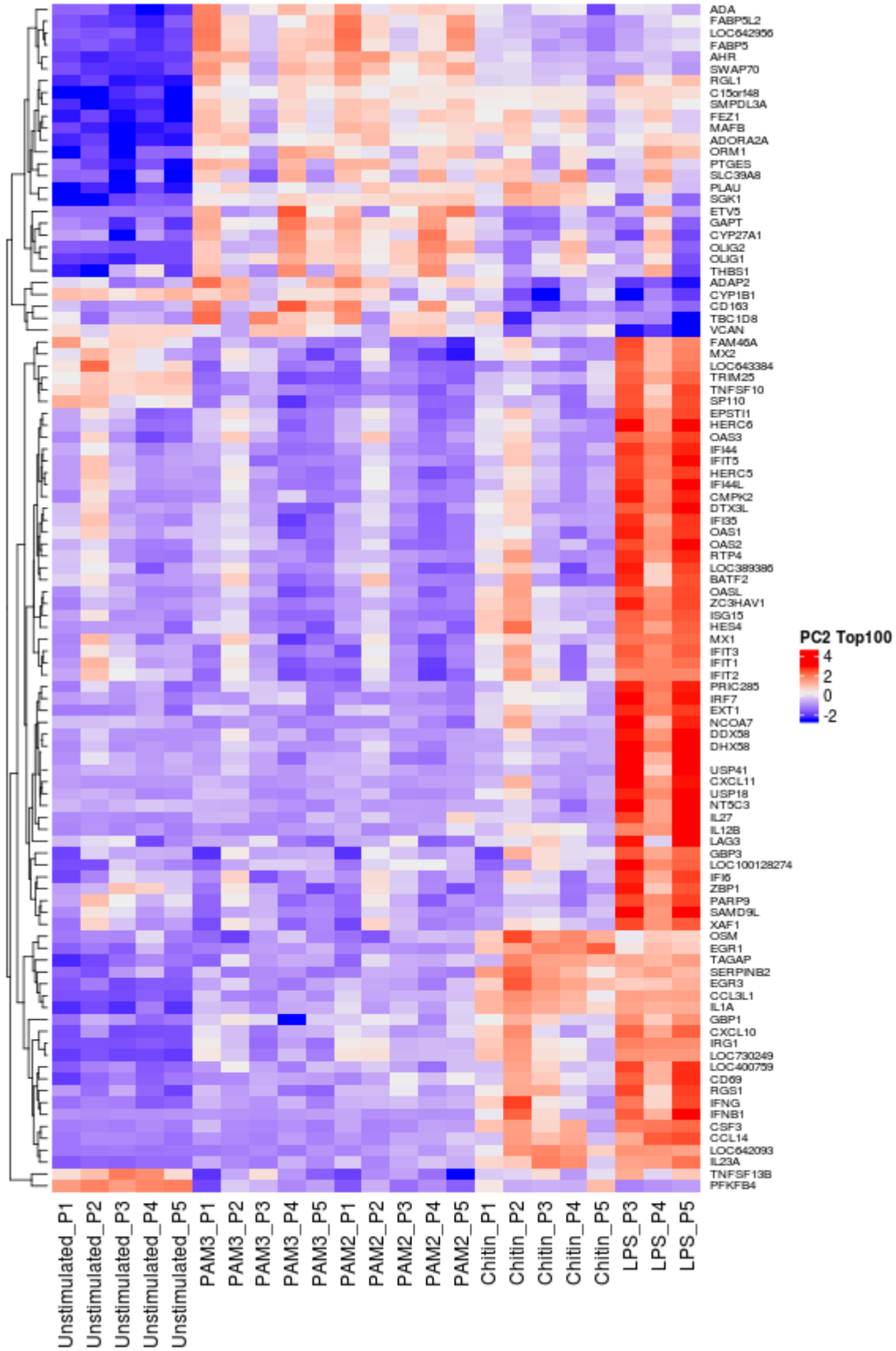
We have used PCA to illustrate treatment differences. Here, we place the PCs into biological context by showing the expression of the 100 genes contributing most to PC1 and PC2.

```
hm.pc1 <- heatmap.E[ match(PCtopDF$PC1_ProbeID[1:100], rownames(heatmap.E)), ]
rownames(hm.pc1) <- hm.pc1$Symbol; hm.pc1$Symbol <- c()
hm.pc1 <- data.frame(t(scale(t(hm.pc1))) )
Heatmap(hm.pc1, cluster_columns = F, cluster_rows=T, name="PC1 Top100",
        row_names_gp = gpar(fontsize=8))
```


SRC
CD83
DRAM1
IRAK2
SGPP2
EBI3
SMPDL3A
ZC3H12A
ADORA2A
RGL1
IL4I1
CD80
BATF
NFKB2
C15orf48
TNFAIP6
IRAK3
PI3
NFKBIZ
SLC11A2
PIM2
PLAU
FSCN1
FEZ1
MAFB
PTGES
PHLDA1
LAMB3
CCL2
CCL7
SERPINB9
ORM1
ORM2
HAMP
SOCS1
ATF3
MIR155HG
TNIP3
CCL8
IRG1
LOC730249
NLF2
INDO
IDO1
CH25H
CXCL10
CFB
CD274
LAMP3
DUSP5
AK3L1
BATF3
SLAMF7
GCH1
MAFF
EGR3
FOSB
TAGAP
SERPINB2
C7orf16
GJB2
CCL23
CXCL2
CXCL1
IL23A
TRAF1
CCRL2
LRRC50
IL1B
CCL3L3
LOC728835
CCL4L2
PPP1R15A
IL1RN
GPR84
ATP5G1
HNRNPA3P1
CCL3
CCL3L1
IL6
SOCS3
SLC39A8
IER3
G0S2
IL8
SLC25A24
PRAM1
NFE2
CREB5
CD300LF
MS4A6A
MOSC1
CHST13
PFKFB4
FRAT2
CXCR1
IL8RB
CD1D
GPBAR1
ASGR1
PC1 Top100
2
1
0
-1
-2
-3
Unstimulated_P1
Unstimulated_P2
Unstimulated_P3
Unstimulated_P4
Unstimulated_P5
PAM3_P1
PAM3_P2
PAM3_P3
PAM3_P4
PAM3_P5
PAM2_P1
PAM2_P2
PAM2_P3
PAM2_P4
PAM2_P5
Chitin_P1
Chitin_P2
Chitin_P3
Chitin_P4
Chitin_P5
LPS_P3
LPS_P4
LPS_P5

In above heatmap, we observe again what we already saw in the logFC-plot above: most genes show the same qualitative reaction (up or down) for all treatments, but differ by how much (e.g. LPS up/downregulates the most).

```
hm.pc2 <- heatmap.E[ match(PCtopDF$PC2_ProbeID[1:100], rownames(heatmap.E)), ]
rownames(hm.pc2) <- hm.pc2$Symbol; hm.pc2$Symbol <- c()
hm.pc2 <- data.frame(t(scale(t(hm.pc2))) )
Heatmap(hm.pc2, cluster_columns = F, cluster_rows=T, name="PC2 Top100",
        row_names_gp = gpar(fontsize=8))
```


ADA
FABP5L2
LOC642956
FABP5
AHR
SWAP70
RGL1
C15or48
SMPDL3A
FEZ1
MAFB
ADORA2A
ORM1
PTGES
SLC39A8
PLAU
SGK1
ETV5
GAPT
CYP27A1
OLIG2
OLIG1
THBS1
ADAP2
CYP1B1
CD163
TBC1D8
VCAN
FAM46A
MX2
LOC643384
TRIM25
TNFSF10
SP110
EPSTI1
HERC6
OAS3
IFI44
IFIT5
HERC5
IFI44L
CMPK2
DTX3L
IFI35
OAS1
OAS2
RTP4
LOC389386
BATF2
OASL
ZC3HAV1
ISG15
HES4
MX1
IFIT3
IFIT1
IFIT2
PRIC285
IRF7
EXT1
NCOA7
DDX58
DHX58
USP41
CXCL11
USP18
NT5C3
IL27
IL12B
LAG3
GBP3
LOC100128274
IFI6
ZBP1
PARP9
SAMD9L
XAF1
OSM
EGR1
TAGAP
SERPINB2
EGR3
CCL3L1
IL1A
GBP1
CXCL10
IRG1
LOC730249
LOC400759
CD69
RGS1
IFNG
IFNB1
CSF3
CCL14
LOC642093
IL23A
TNFSF13B
PFKFB4
PC2 Top100
4
2
0
-2
Unstimulated_P1
Unstimulated_P2
Unstimulated_P3
Unstimulated_P4
Unstimulated_P5
PAM3_P1
PAM3_P2
PAM3_P3
PAM3_P4
PAM3_P5
PAM2_P1
PAM2_P2
PAM2_P3
PAM2_P4
PAM2_P5
Chitin_P1
Chitin_P2
Chitin_P3
Chitin_P4
Chitin_P5
LPS_P3
LPS_P4
LPS_P5

For PC2, we see that LPS has many additional genes upregulated which are almost unchanged in the other treatments. Also, Chitin sometimes behaves more like LPS, and sometimes more like PAMs. We also remark that donor P2 is behaving somewhat as an outlier.

```
#
```

End of Script

```
sessionInfo()
```

```
## R version 3.4.2 (2017-09-28)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.5 LTS
##
## Matrix products: default
## BLAS: /usr/lib/libblas/libblas.so.3.0
## LAPACK: /usr/lib/lapack/liblapack.so.3.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] ComplexHeatmap_1.17.1 pheatmap_1.0.8        forcats_0.2.0
##  [4] stringr_1.3.0         dplyr_0.7.4           purrr_0.2.4
##  [7] readr_1.1.1           tidyr_0.8.0           tibble_1.4.2
## [10] tidyverse_1.2.1       cowplot_0.9.2         ggplot2_2.2.1
## [13] limma_3.34.8
##
## loaded via a namespace (and not attached):
##  [1] statmod_1.4.30       shape_1.4.4          circlize_0.4.3
##  [4] tidyselect_0.2.3     GetoptLong_0.1.6     reshape2_1.4.3
##  [7] splines_3.4.2        haven_1.1.1          lattice_0.20-35
## [10] colorspace_1.3-2     htmltools_0.3.6      yaml_2.1.16
## [13] rlang_0.1.6          pillar_1.1.0         foreign_0.8-69
## [16] glue_1.2.0           RColorBrewer_1.1-2   modelr_0.1.1
## [19] readxl_1.0.0         bindrcpp_0.2         bindr_0.1
## [22] plyr_1.8.4           munsell_0.4.3        gtable_0.2.0
## [25] cellranger_1.1.0     rvest_0.3.2          GlobalOptions_0.0.12
## [28] psych_1.7.8          evaluate_0.10.1      labeling_0.3
## [31] knitr_1.20           parallel_3.4.2       broom_0.4.3
## [34] Rcpp_0.12.15         scales_0.5.0         backports_1.1.2
## [37] jsonlite_1.5         mnormt_1.5-5         rjson_0.2.15
```

```
## [40] hms_0.4.1           digest_0.6.15       stringi_1.1.6
## [43] rprojroot_1.3-2     cli_1.0.0           tools_3.4.2
## [46] magrittr_1.5        lazyeval_0.2.1      crayon_1.3.4
## [49] pkgconfig_2.0.1     xml2_1.2.0          lubridate_1.7.2
## [52] assertthat_0.2.0    rmarkdown_1.8.10    httr_1.3.1
## [55] rstudioapi_0.7      R6_2.2.2            nlme_3.1-131
## [58] compiler_3.4.2
```